To appear in IEEE Transactions on Image Processing, 2016

LSDT: Latent Sparse Domain Transfer Learning for Visual Adaptation

Lei Zhang, *Member, IEEE*, Wangmeng Zuo, *Senior Member, IEEE*, and David Zhang, *Fellow, IEEE*

***Abstract*—We propose a novel reconstruction based transfer learning method called Latent Sparse Domain Transfer (LSDT) for domain adaptation and visual categorization of heterogeneous data. For handling cross-domain distribution mismatch, we advocate reconstructing the *target domain* data with the combined *source* and *target domain* data points based on ℓ_1 -norm sparse coding. Furthermore, we propose a joint learning model for simultaneous optimization of the sparse coding and the optimal subspace representation. Additionally, we generalize the proposed LSDT model into a kernel based linear/nonlinear basis transformation learning framework for tackling nonlinear subspace shifts in Reproduced Kernel Hilbert Space. The proposed methods have three advantages: 1) the latent space and reconstruction are jointly learned for pursuit of an optimal subspace transfer; 2) with the theory of sparse subspace clustering (SSC), a few valuable source and target data points are formulated to reconstruct the target data with noise (outliers) from source domain removed during domain adaptation, such that the robustness is guaranteed; 3) a nonlinear projection of some latent space with kernel is easily generalized for dealing with highly nonlinear domain shift (e.g. face poses). Extensive experiments on several benchmark vision datasets demonstrate that the proposed approaches outperform other state-of-the-art representation based domain adaptation methods.**

***Index Terms*—Transfer learning, domain adaptation, visual categorization, heterogeneous data**

I. Introduction

Visual big data bring many challenges to machine learning and computer vision, e.g. the dilemma of insufficient labeled data. One interesting topic is to enrich the limited labeled data with relevant data from web or other sources and exploit the unlabeled data by semi-supervised learning (SSL) [31, 32]. However, the enriched data from *target* domain is violated from the training data in *source* domain [33], which leads to significant performance degradation in classification [7]. Domain adaptation, that has the same goal as transfer learning, aims at transferring knowledge across different but related domains, i.e. $P(\mathbf{X}_S|\mathbf{Y}_S) \neq P(\mathbf{X}_T|\mathbf{Y}_T)$ [34, 35], where ($\mathbf{X}_S$, $\mathbf{Y}_S$) denote the source data matrix and the corresponding label matrix, ($\mathbf{X}_T$, $\mathbf{Y}_T$) represent the target data matrix and label matrix. Physically, such subspace mismatch or domain shift/bias is common in vision problems. It often results from a variety of visual cues or abrupt feature changes, such as camera viewpoint, resolution, illumination, color, poses, and background, etc. To this end, various domain adaptation methods have been developed to adapt a model from source to target domain, including representation-based and classifier-based ones. The former tends to achieve domain alignment by learning a transformation [8, 14, 15, 19]. The latter advocates learning a robust classifier with $\mathbf{X}_S$ and $\mathbf{X}_T$ by introducing some ad-hoc regularization [11, 16, 17, 40]. The common practice is to train a classifier on source data and find an optimal decision boundary on both domains.

In this paper, we focus on reconstruction based domain adaptation via latent subspace learning and sparse representation. Recently, a low-rank representation (LRR) based domain adaptation framework has been proposed for knowledge transfer, i.e. RDALR [2] and LTSL [1]. The basic idea of RDALR is illustrated in Fig. 1(a). A rotation $\mathbf{W}$ is used to transform the source data $\mathbf{X}_S$, then do alignment by reconstructing the rotated source data via LRR. However, finding such an alignment between $\mathbf{W}\mathbf{X}_S$ and $\mathbf{X}_T$ may not transfer knowledge directly and it is unclear if a test sample is from the source domain or the target. Fig. 1(b) illustrates the basic idea of LTSL, where the subspace projection $\mathbf{W}$ is pre-learned by using PCA, LDA, etc. Then, the projected source data $\mathbf{W}\mathbf{X}_S$ is used to reconstruct the projected target data $\mathbf{W}\mathbf{X}_T$ via LRR. Both methods are inadequate in knowledge transfer and subspace alignment, with three reasons as follows.

First, in LTSL the subspace is pre-learned and is independent with the reconstruction process, which limits the domain adaptation performance. Therefore, we propose a joint learning method for the pursuit of the latent subspace $\mathbf{P}$ and reconstruction $\mathbf{Z}$. The joint learning of $\mathbf{P}$ and $\mathbf{Z}$ makes our method distinctly different with RDALR [2] and LTSL [1] in both model and algorithm. Experiments on face and object datasets show that joint learning improves the recognition accuracy by 3% and 17%, respectively.

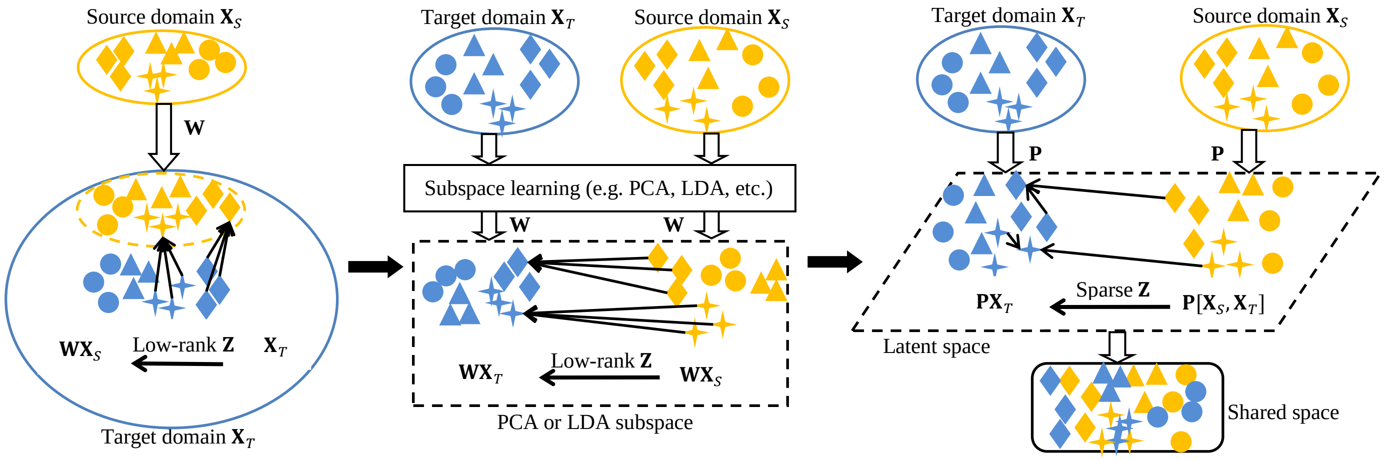
Second, in both RDALR and LTSL, the data in target domain are reconstructed with the data in source domain only by using LRR [4]. Two noteworthy things include: (i) LRR was suggested to get the block diagonal solution for subspace segmentation. However, trivial solution will be obtained when handling the disjoint subspace and insufficient data. Moreover, LRR based domain adaptation is with a strong independent

This work was supported by National Natural Science Foundation of China (No. 61401048), Research funds for Central Universities, and the Hong Kong Scholar Program (No.XJ2013044).

● L. Zhang is with College of Communication Engineering, Chongqing University, Chongqing, China and Department of Computing, The Hong Kong Polytechnic University, Hong Kong. (e-mail: leizhang@cqu.edu.cn).

● W.M. Zuo is with the Harbin Institute of Technology, Harbin, China. (e-mail: cswmzuo@gmail.com)

● D. Zhang is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong (e-mail: csdzhang@comp.polyu.edu.hk).


Source domain $\mathbf{X}_S$
W
$\mathbf{WX}_S$
Low-rank **Z**
$\mathbf{X}_T$
Target domain $\mathbf{X}_T$
Target domain $\mathbf{X}_T$
Source domain $\mathbf{X}_S$
Subspace learning (e.g. PCA, LDA, etc.)
W
W
$\mathbf{WX}_T$
Low-rank **Z**
$\mathbf{WX}_S$
PCA or LDA subspace
Target domain $\mathbf{X}_T$
Source domain $\mathbf{X}_S$
P
P
$\mathbf{PX}_T$
Sparse **Z**
$\mathbf{P}[\mathbf{X}_S, \mathbf{X}_T]$
Latent space
Shared space

(a) Idea of RDALR (b) Idea of LTSL (c) Idea of our method

Fig. 1. Overview of the existing reconstruction guided knowledge transfer methods and our method.

subspace assumption. Different from LRR, sparse subspace clustering (SSC) [3, 21, 37] is for data points that lie in a union of low-dimensional subspaces, where a sparse matrix **Z** is learned by minimizing $\|\mathbf{X}-\mathbf{XZ}\|_{\mathrm{F}}$. Compared with LRR, SSC can be scalable [43], is well supported by both theoretical analysis [37] and experimental results [3] in handling the data points near the intersections of subspaces. Therefore, in light of the multi-source data lying in different space, we are inspired to reconstruct the target data $\mathbf{X}_T$ with the source data $\mathbf{X}_S$ by learning a sparse **Z** based on SSC theory. With face and object datasets, an increment of 2% and 6.4% recognition accuracy is achieved by using SSC-based reconstruction. (ii) For RDALR and LTSL, the target data $\mathbf{X}_T$ are reconstructed by solely using the source data $\mathbf{X}_S$. When only very few source data is available, better reconstruction can be obtained by grouping the target data i.e. $\mathbf{X}=\left[\mathbf{X}_{S,}\mathbf{X}_T\right]$ as “dictionary”. Fortunately, with the SSC theory [37], we can use both $\mathbf{X}_S$ and $\mathbf{X}_T$ for reconstructing the target data and avoid the trivial solution. The experiments on face and object datasets demonstrate that 4.7% and 9.7% increments of recognition accuracy are achieved by comparing with that of using source data only.

Third, the existing methods work as a linear framework, and cannot tackle the nonlinear shifts in real-world vision problems. Therefore, it is valuable to develop a nonlinear reconstruction guided subspace transfer framework. In this work, we generalize our model to tackle nonlinear shifts in Reproduced Kernel Hilbert Space. The experiments on face and object datasets demonstrate that our method is 7.7% and 6.3% higher than linear ones in recognition accuracy, respectively.

In this paper, following the subspace reconstruction guided domain adaptation framework, we propose a sparse reconstruction method in the learned latent space between the source data $\mathbf{X}_S$ and the target data $\mathbf{X}_T$. It tries to account for noise in data corruption and removes outliers, with their intrinsic relatedness preserved. More formally, we name the proposed method as latent sparse domain transfer (LSDT), which aims to learn a sparse reconstruction coefficient matrix between domains in some latent space for domain adaptation. The basic idea of LSDT is illustrated in Fig. 1(c). Compared with RDALR and LTSL, our LSDT can jointly learn the latent space **P** and the SSC-based reconstruction **Z**, and the target data $\mathbf{X}_T$ is reconstructed with the group data of $\mathbf{X}_S$ and $\mathbf{X}_T$, such that the source and target data lie in a shared latent space with domain shift/bias removed.

In summary, the key contributions of this work are threefold.

- The latent space projection **P** and the sparse reconstruction coefficient matrix **Z** are simultaneously learnt via a joint learning mechanism, which can achieve an optimal subspace representation. The sparse property implies that only a few data points from source domain are selected for subspace transfer and overcomes the overfitting problem.
- The sparse subspace clustering (SSC) is introduced for reconstruction guided domain adaptation. The combined source and target data are used to reconstruct the target domain, which can better span the entire feature space than the under-complete source data only. In particular, the trivial solution can be avoided by using SSC instead of LRR.
- Induced by Mercer kernel theorem, the proposed method is generalized as a nonlinear method, in which the domain adaptation is employed in a reproduced kernel Hilbert space (RKHS) for handling nonlinear domain shift.

Paper organization. This paper is organized as follows. In Section 2, we give a brief overview of the related work in domain adaptation. The proposed latent sparse domain transfer method is illustrated in Section 3. The proposed nonlinear LSDT is presented in Section 4. The experimental results for several domain adaptation based vision tasks are shown in Section 5. The in-depth discussion of the proposed methods is illustrated in Section 6. Section 7 concludes the paper.

II. Related Works

Domain adaptation can be performed in either representation level or classifier level [9, 10, 12, 14, 15, 16, 17, 36]. In classifier based adaptation, Yang *et al.* [12] proposed an adaptive SVM (ASVM) where the source classifier $f^S(\mathbf{x})$ was adapted to the target classifier $f^T(\mathbf{x})$ by learning a perturbation $\Delta f(\mathbf{x})$, such that $f^T(\mathbf{x}) = f^S(\mathbf{x}) + \Delta f(\mathbf{x})$. Similarly, Duan *et al* [16] proposed an adaptive multiple kernel learning (AMKL) for consumer video event recognition from annotated web videos. Zhang *et al.* [36] proposed a DA framework with two error terms based on ℓ_2-norm regularization. However, for classifier-based methods, the label information of source and target domains should be used for learning a target classifier.

To learn a better data representation for adaptation without labels used, Gong *et al.* [10] proposed an unsupervised domain adaptation method (GFK), in which geodesic flow kernel is used to model the domain shift by integrating an infinite number of subspaces where the changes in geometric and statistical properties are characterized. Gopalan *et al.* [8] also proposed an unsupervised method (SGF) for low dimensional subspace transfer. The idea behind SGF is that it samples a group of subspaces along the geodesic between source and target data, and project the source data into the subspaces for discriminative classifier learning. Shekhar *et al.* [6] proposed a shared domain dictionary learning (SDDL), which assumes that the knowledge of two domains can be integrated into one dictionary **D**. However, the label information of source and target data is still required, while the proposed method does not need the label information during cross-domain learning. In [44], Lin *et al.* proposed a dynamic spatio-temporal subspace i.e. STDM, for background subtraction, where incremental subspace learning and analytical linear reconstruction are used to maintain the dynamic space.

In reconstruction based adaptation, RDALR and LTSL that are most structurally relevant with this paper were proposed by Jhuo *et al.* [2] and Shao *et al.* [1], respectively, in which low rank representation (LRR) is used for subspace transfer. A brief overview of RDALR and LTSL is introduced as follows.

A. *Robust Domain Adaptation via Low Rank (RDALR)* [2]

RDALR shown in Fig. 1(a) addresses the domain adaptation problem by minimizing the following objective function

$$\begin{aligned}&\min_{\mathbf{W},\mathbf{Z},\mathbf{E}} rank(\mathbf{Z})+\alpha\|\mathbf{E}\|_{2,1}\\ &\text{s.t.}\ \mathbf{W}\mathbf{X}_S=\mathbf{X}_T\mathbf{Z}+\mathbf{E},\ \mathbf{W}\mathbf{W}^{\mathrm{T}}=\mathbf{I}\end{aligned} \tag{1}$$

where *rank*(·) represents the rank of a matrix, $\|\mathbf{E}\|_{2,1}$ denotes $\ell_{2,1}$ -norm, and α is the regularization coefficient. The constraint $\mathbf{W}\mathbf{W}^{\mathrm{T}}=\mathbf{I}$ is introduced to learn an orthogonal transformation matrix. The term $\|\mathbf{E}\|_{2,1}$ is used to encourage the error columns of **E** to be 0, such that noise or outliers in source domain can be removed during adaptation. While minimization of *rank*(**Z**) tends to find a reconstruction coefficient matrix with the lowest rank structure. In optimization, due to the discrete nature of rank function, nuclear norm or trace norm (i.e. the sum of singular values of the matrix) is generally adopted as a proper surrogate of the rank. Then, inexact Augmented Lagrange Multiplier (ALM) [22] can be used for solving problem (1).

B. *Low-rank Transfer Subspace Learning (LTSL)* [1]

Similarly but different in nature from RADLR, LTSL shown in Fig. 1(b) addresses the subspace transfer problem by minimizing the following objective function

$$\begin{aligned}&\min_{\mathbf{W},\mathbf{Z},\mathbf{E}} F(\mathbf{W},\mathbf{X}_S)+\lambda_1 rank(\mathbf{Z})+\lambda_2\|\mathbf{E}\|_{2,1}\\ &\text{s.t.}\ \mathbf{W}^{\mathrm{T}}\mathbf{X}_T=\mathbf{W}^{\mathrm{T}}\mathbf{X}_S\mathbf{Z}+\mathbf{E},\ \mathbf{W}^{\mathrm{T}}\mathbf{U}_2\mathbf{W}=\mathbf{I}\end{aligned} \tag{2}$$

where $F(\mathbf{W},\mathbf{X}_S)$ is a generalized subspace learning function which can be written as $Tr(\mathbf{W}^{\mathrm{T}}\mathbf{U_1}\mathbf{W})$, $\mathbf{U_1}$ and $\mathbf{U_2}$ are selected based on the conventional subspace learning model, such as PCA, LDA, etc. Given fixed $\mathbf{W}$, inexact ALM under convex surrogate of rank function can be used to solve problem (2), which is similar to (1). There are three main differences between LTSL and RDALR:

- RDALR tends to reconstruct the rotated source data $\mathbf{X}_S$ by using target data $\mathbf{X}_T$. While LTSL attempts to reconstruct the target data using the source data in the learned subspace.
- RDALR first use **W** to rotate the source data, and perform the data alignment in the original space of target data. While LTSL aims to find a subspace alignment between $\mathbf{X}_S$ and $\mathbf{X}_T$.
- A subspace learning function is embedded into LTSL for learning a transformation **W** with discriminative property.

In summary, both RDALR and LTSL perform the domain adaptation using LRR. The former presents to data alignment by leveraging LRR and provides some valuable insight for domain adaptation. LTSL performs adaptation in some pre-learned subspace, and presents a more complete theoretical and subspace analysis for knowledge adaptation. As mentioned, Liu *et al.* [4, 21] proved that LRR performs well when the subspaces are independent and the data sampling is sufficient.

However, this assumption is difficult to hold in cross-domain vision problems (i.e. data distribution mismatch). Following the representation based adaptation, our proposed method attempts to use SSC based sparse reconstruction for subspace transfer while avoiding such strong low-rank assumption. More advantageously, the proposed method can simultaneously learn a linear/nonlinear basis transformation for subspace projection and a sparse reconstruction matrix with stronger robustness. It can prohibit the noise or outliers in source domain from transferring to target domain and also avoid overfiting in reconstruction, especially when the number of source data and target data is not sufficient. The proposed method is different from LTSL in three aspects. (i) The joint learning of subspace and reconstruction. (ii) Sparse reconstruction based on SSC using combined source and target data. (iii) Kernel based nonlinear domain adaptation. Fig. 2 illustrates the flowchart of the proposed method for heterogeneous image classification.

III. Latent Sparse Domain Transfer Learning

A. *Notations*

In this paper, the source and target domain are defined by subscript "*S*" and "*T*", respectively. The training data of source and target domain is denoted as $\mathbf{X}_S\in\Re^{D\times N_S}$ and $\mathbf{X}_T\in\Re^{D\times N_T}$, respectively, where D is the number of dimensions, N_S and N_T are the number of training samples in both domains. $\mathbf{X}_{Tl}\in\Re^{D\times N_{Tl}}$ and $\mathbf{X}_{Tu}\in\Re^{D\times N_{Tu}}$ denote the few labeled and most unlabeled data of target domain. Let $\mathbf{P}\in\Re^{d\times D}$ represents a basis transformation. The sparse reconstruction matrix between $\mathbf{X}_S$ and $\mathbf{X}_T$ is denoted as **Z**. $\mathbf{1}_n$ denotes a full-one column vector with length of n and **I** denotes an identity matrix. $\|\cdot\|_0$ counts the number of nonzero elements of a vector, $\|\cdot\|_p$ (p = 0, 1 or 2) denotes ℓ_p-norm, and $\|\cdot\|_{\mathrm{F}}$ denote Frobenius norm of a matrix. $[\mathbf{X}]_i$ denotes the *i*-th column of **X**. Note that matrix and

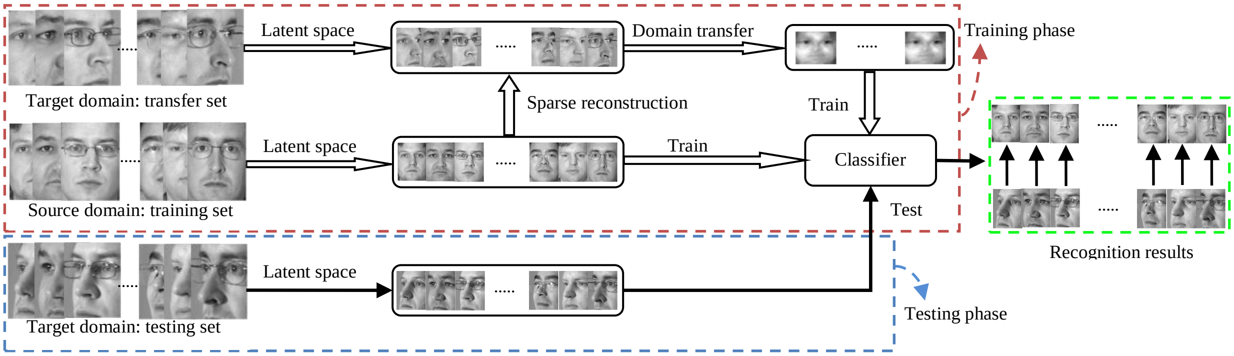


Fig. 2. Flowchart of the training and testing phase of the proposed LSDT method for visual categorization.

vector is in capital and lower bold face, and variable is in italics.

B. Problem Formulation

As illustrated in Fig. 1, we aim to learn a reconstruction coefficient matrix **Z** for representing target data $\mathbf{X}_T$ by using itself and source data $\mathbf{X}_S$ together in some latent space projected by a pre-defined basis transformation **P**. Therefore, the optimization problem can be formulated as

$$\min_{\mathbf{Z}} \left\|\mathbf{P}\mathbf{X}_T - \mathbf{P}[\mathbf{X}_S, \mathbf{X}_T]\mathbf{Z}\right\|_F^2, \quad \text{s.t.} \|\mathbf{Z}\|_0 \le T_0 \tag{3}$$

where $\mathbf{Z} \in \Re^{(N_S+N_T)\times N_T}$, T_0 is the sparsity level. Due to that ℓ_0-norm based optimization is non-convex, in this paper, ℓ_1-norm is used in the proposed model.

For learning such a basis transformation **P** which can ensure that the projection does not distort the data and can remain too much available information, the following term is integrated,

$$\min_{\mathbf{P}} \left\|[\mathbf{X}_S, \mathbf{X}_T] - \mathbf{P}^{\mathrm{T}}\mathbf{P}[\mathbf{X}_S, \mathbf{X}_T]\right\|_F^2 \tag{4}$$

By combining (3) and (4) together, the final formulation of the proposed LSDT method is represented as follows

$$\min_{\mathbf{Z},\mathbf{P}} \|\mathbf{Z}\|_1 + \lambda_1 \left\|\mathbf{P}\mathbf{X}_T - \mathbf{P}[\mathbf{X}_S, \mathbf{X}_T]\mathbf{Z}\right\|_F^2 + \lambda_2 \left\|[\mathbf{X}_S, \mathbf{X}_T] - \mathbf{P}^{\mathrm{T}}\mathbf{P}[\mathbf{X}_S, \mathbf{X}_T]\right\|_F^2 \tag{5}$$
$$\text{s.t. } \mathbf{P}\mathbf{P}^{\mathrm{T}} = \mathbf{I}, \mathbf{1}_{N_S+N_T}^{\mathrm{T}}\mathbf{Z} = \mathbf{1}_{N_T}^{\mathrm{T}}, Z_{N_S+i,i} = 0, \forall i = 1, \cdots, N_T$$

where the rows of **P** are required to be orthogonal and normalized to unit norm for preventing the solution degenerate into zero by enforcing $\mathbf{P}\mathbf{P}^{\mathrm{T}} = \mathbf{I}$. Additionally, we also impose $\mathbf{1}_{N_S+N_T}^{\mathrm{T}}\mathbf{Z} = \mathbf{1}_{N_T}^{\mathrm{T}}$ for addressing the problem that source and target data lie in a union of affine subspaces instead of linear subspaces. λ_1 and λ_2 denote the tradeoff parameters.

For simplification, we let $\mathbf{X} = [\mathbf{X}_S, \mathbf{X}_T] \in \Re^{D\times N}$ then the objective function of problem (5) can be written as

$$J_1(\mathbf{P}, \mathbf{Z}, \mathbf{X}_T, \mathbf{X}) = \|\mathbf{Z}\|_1 + \lambda_1 \|\mathbf{P}\mathbf{X}_T - \mathbf{P}\mathbf{X}\mathbf{Z}\|_F^2 + \lambda_2 \left\|\mathbf{X} - \mathbf{P}^{\mathrm{T}}\mathbf{P}\mathbf{X}\right\|_F^2 \tag{6}$$

One proposition on the basis transformation **P** is as follows.

Proposition 1. There exists an optimal solution $\mathbf{P}^*$ that can be intuitively represented as a linear combination of raw source and target data **X** for some $\mathbf{\Phi} \in \Re^{N\times d}$ in the following form

$$\mathbf{P}^* = \mathbf{\Phi}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}} \tag{7}$$

Note that Proposition 1 has also been used in subspace clustering and dictionary learning [6, 13]. With Proposition 1, by substituting (7) into (6), the objective function is written as

$$J_2(\mathbf{\Phi}, \mathbf{Z}, \mathbf{X}_T, \mathbf{X}) = \|\mathbf{Z}\|_1 + \lambda_1 \left\|\mathbf{\Phi}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X}_T - \mathbf{\Phi}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{Z}\right\|_F^2 + \lambda_2 \left\|\mathbf{X} - \mathbf{X}\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X}\right\|_F^2 \tag{8}$$

Let $\mathbf{K}_T = \mathbf{X}^{\mathrm{T}}\mathbf{X}_T$, $\mathbf{K} = \mathbf{X}^{\mathrm{T}}\mathbf{X}$, then the proposed method (5) can be illustrated as follows

$$\min_{\mathbf{Z},\mathbf{\Phi}} \|\mathbf{Z}\|_1 + \lambda_1 \left\|\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}_T - \mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\mathbf{Z}\right\|_F^2 + \lambda_2 \left\|\mathbf{X} - \mathbf{X}\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\right\|_F^2 \tag{9}$$
$$\text{s.t. } \mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\mathbf{\Phi} = \mathbf{I}, \mathbf{1}_{N_S+N_T}^{\mathrm{T}}\mathbf{Z} = \mathbf{1}_{N_T}^{\mathrm{T}}, Z_{N_S+i,i} = 0 \,\forall i = 1, \cdots, N_T$$

From (9), it is observed that a nonlinear framework of LSDT can be deducted by using a nonlinear mapping function φ. The details can be referred as Section IV.

For our LSDT model in Eq. (5), when fixed **P**, the sub-problem on **Z** shares similar formulation with SSC [3] and RSC [37]. Based on the theoretical results in [37], our LSDT model is also feasible in recovering the underlying subspace structures. However, the model in Eq. (5) is non-convex, making it difficult to extend the theoretical results [37] to the full LSDT model.

C. Optimization

It can be seen from problem (9) that two variables are involved. To solve this minimization, alternative optimization strategy that solve one variable while fixing the other one is considered. Therefore, two main steps are included.

- **Update Z:**

For solving **Z**, one can fix **Φ**, then the minimization problem (9) with respect to **Z** becomes

$$\min_{\mathbf{Z}} \|\mathbf{Z}\|_1 + \lambda_1 \left\|\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}_T - \mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\mathbf{Z}\right\|_F^2 \tag{10}$$
$$\text{s.t. } \mathbf{1}_{N_S+N_T}^{\mathrm{T}}\mathbf{Z} = \mathbf{1}_{N_T}^{\mathrm{T}}, Z_{N_S+i,i} = 0, \forall i = 1, \cdots, N_T$$

This is a typical sparse Lasso optimization problem with linear equality constraints, and can be efficiently solved by using alternative direction multiplier method (ADMM) in [3]. A full description of ADMM can be referred as [26] for interested readers. The solving process of problem (10) by using ADMM is outlined in Algorithm 1. The deduction for solving **Z** can be found in Appendix A.

- **Update Φ:**

For solving **Φ**, the minimization problem (9) after fixing **Z** can be written as

$$\min_{\mathbf{\Phi}} \lambda_1 \left\|\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}_T - \mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\mathbf{Z}\right\|_F^2 + \lambda_2 \left\|\mathbf{X} - \mathbf{X}\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\right\|_F^2 \tag{11}$$
$$\text{s.t. } \mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\mathbf{\Phi} = \mathbf{I}$$

Algorithm 1. Solving Problem (10) by ADMM

```
Input: $\lambda_1$, $\mathbf{K}_T \in \Re^{N\times N_S}$, $\mathbf{K} \in \Re^{N\times N}$, $\mathbf{\Phi}$, $\rho = 1.1$, and $\max_\mu = 10^6$
Initialize: $\mathbf{Z}$=0, $\mathbf{Y}_A$=0, $\mathbf{Y}_B$=0, $\mathbf{Y}_C$=0, $\mathbf{Y}_D$=0, $\mu_2=\lambda_1$
  while not converge do
    1. Fix Z and U, and update L by
       $\mathbf{L} = \left(\mu_1\mathbf{K}^{\mathrm{T}}\mathbf{\Phi\Phi}^{\mathrm{T}}\mathbf{K} + \mu_2\mathbf{I} + \mu_2\mathbf{1}_{N_S+N_T}\mathbf{1}_{N_S+N_T}^{\mathrm{T}}\right)^{-1}$
       $\left(\mu_1\mathbf{K}^{\mathrm{T}}\mathbf{\Phi\Phi}^{\mathrm{T}}\mathbf{K}_T - \mathbf{Y}_A - \mathbf{1}_{N_S+N_T}\mathbf{Y}_B + \mu_2\mathbf{Z} + \mu_2\mathbf{1}_{N_S+N_T}\mathbf{1}_{N_T}^{\mathrm{T}}\right)$
    2. Fix Z and L, and update U by
       $U_i = \dfrac{\mu_2 Z_{N_S+i,i} + Y_C^i - Y_D^i}{2\mu_2}, i = 1,\ldots,N_T$
    3. Fix L and U, and update Z as follows
    3.1. for $Z_{N_S+i,i}$ $\forall i = 1,\cdots,N_T$, there is
       $\min\left|Z_{N_S+i,i}\right| + \mu_2\left(Z_{N_S+i,i} - \left(\dfrac{L_{N_S+i,i}+U_i}{2} + \dfrac{Y_A^{N_S+i,i}+Y_C^i}{2\mu_2}\right)\right)$
    3.2. for Z other than $Z_{N_S+i,i}$, there is
       $\min_{\mathbf{Z}} \|\mathbf{Z}\|_1 + \dfrac{\mu_2}{2}\left\|\mathbf{Z} - \left(\mathbf{L} + \dfrac{\mathbf{Y}_A}{\mu_2}\right)\right\|_{\mathrm{F}}^2$
    4. Update the multipliers
       $\mathbf{Y}_A = \mathbf{Y}_A + \mu_2(\mathbf{L} - \mathbf{Z})$
       $\mathbf{Y}_B = \mathbf{Y}_B + \mu_2(\mathbf{1}^{\mathrm{T}}\mathbf{L} - \mathbf{1}^{\mathrm{T}})$
       $\mathbf{Y}_C^i = \mathbf{Y}_C^i + \mu_2(Z_{N_S+i,i} - U_i)$
       $\mathbf{Y}_D = \mathbf{Y}_D + \mu_2\mathbf{U}$
    5. Update the parameter $\mu_2$
       $\mu_2 = \min(\mu_2\rho,\ \max_\mu)$
    6. Check the convergence
  end while
Output: Z
```

We have the following proposition for solving $\mathbf{\Phi}$ in (11).

Proposition 2. When $\mathbf{Z}$ is fixed, the optimal solution of (11) is computed as

$$\mathbf{\Phi}^* = \mathbf{VS}^{-\frac{1}{2}}\mathbf{\Omega}^*$$

where $\mathbf{V}$ and $\mathbf{S}$ are from the eigen-decomposition of $\mathbf{K} = \mathbf{VSV}^{\mathrm{T}}$, and $\mathbf{\Omega}^*$is the optimal solution of the following problem

$$\mathbf{\Omega}^* = \arg\min_{\mathbf{\Omega}} Tr\left(\mathbf{\Omega}^{\mathrm{T}}\mathbf{\Theta\Omega}\right), \text{ s.t. } \mathbf{\Omega}^{\mathrm{T}}\mathbf{\Omega} = \mathbf{I}$$

The optimization of problem (11) is outlined in Algorithm 2. The deduction of the proposition 2 can be found in Appendix B.

In summary, with the two updating steps for $\mathbf{Z}$ and $\mathbf{\Phi}$ based on Algorithm 1 and Algorithm 2, the complete optimization of the proposed LSDT method is illustrated in Algorithm 3.

D. Remarks on the Convergence

From the viewpoint of optimization, the proposed LSDT is non-convex *w.r.t.* $\mathbf{Z}$ and $\mathbf{\Phi}$, but the global solution of each when fixing the other can be solved. The local optimum of the model can be guaranteed using the proposed optimization method. The convergence is shown in the *Discussion* part (*please* see Fig. 8c). After 5 iterations, a local optimum can be achieved for two datasets, as an example.

From the level of approach, by comparing to LTSL [1], it pre-learns a transformation $\mathbf{P}$ using PCA or LDA, then solves the $\mathbf{Z}$ by using low-rank constraint, such that the performance must be sub-optimal with the pre-learned $\mathbf{P}$ as a warm start without update. To overcome the flaw of such a suboptimal $\mathbf{P}$, The proposed method aims at learning $\mathbf{P}$ and $\mathbf{Z}$ simultaneously by using an alternating optimization strategy, such that better performance can be expected.

Algorithm 2. Solving Problem (11) by Proposition 2

```
Input: $\mathbf{K}_T \in \Re^{N\times N_S}$, $\mathbf{K} \in \Re^{N\times N}$, $\mathbf{Z}$, $\lambda_1$, $\lambda_2$;
Procedure:
  1. Perform eigenvalue decomposition $\mathbf{K} = \mathbf{VSV}^{\mathrm{T}}$
  2. Compute $\mathbf{\Theta} = \mathbf{S}^{\frac{1}{2}}\mathbf{V}^{\mathrm{T}}(\lambda_1(\mathbf{K}^{-1}\mathbf{K}_T - \mathbf{Z})(\mathbf{K}^{-1}\mathbf{K}_T - \mathbf{Z})^{\mathrm{T}} - \lambda_2\mathbf{I})\mathbf{VS}^{\frac{1}{2}}$
  3. Perform eigenvalue decomposition of $\mathbf{\Theta} = \mathbf{U\Sigma U}^{\mathrm{T}}$
  4. Get $\mathbf{\Omega} = \mathbf{U}(:,\mathcal{U})$, where $\mathcal{U}$ is index of the $d$ smallest eigenvalues
  5. Obtain $\mathbf{\Phi} = \mathbf{VS}^{-\frac{1}{2}}\mathbf{\Omega}$;
Output: $\mathbf{\Phi}$
```

E. Computational Complexity

Algorithm 3 includes two steps: update $\mathbf{Z}$ (Algorithm 1) and update $\mathbf{\Phi}$ (Algorithm 2). For Algorithm 1 (i.e. ADMM), suppose that the number of iterations is T_1, the complexity of computing $\mathbf{L}$ is $O(T_1N^3)$ and the complexity of computing $\mathbf{Z}$ is $O(T_1N^2)$. Therefore, the computational complexity of Algorithm 1 is $O(T_1N^3)$+$O(T_1N^2)$. For Algorithm 2, the eigen-decomposition and matrix multiplication are involved, with the computational complexity of $O(N^3)$. Suppose that the number of iterations in Algorithm 3 is T, then the total computational complexity of LSDT can be expressed as $O(TT_1N^3)$+$O(TT_1N^2)$+$O(TN^3)$.

IV. Nonlinear Domain Transfer Learning

A. Formulation of NLSDT

In LSDT, a linear transformation $\mathbf{P}$ is exploited for latent subspace learning. Naturally, NLSDT is a nonlinear extension of LSDT by mapping the data from original space $\Re^D$ to the reproducing kernel Hilbert space (RKHS) $\mathcal{H}$, that is defined as $\varphi: \Re^D \to \mathcal{H}$, induced by Mercer kernel. In RKHS, a nonlinear transformation $\boldsymbol{\mathcal{P}}$ is learned to handle nonlinear domain bias, such as rotation of poses in face recognition.

For introducing the framework of NLSDT, we first define the kernel gram matrix, which is denoted as the matrix $\boldsymbol{\mathcal{K}}$, and $[\boldsymbol{\mathcal{K}}]_{i,j} = \langle\varphi(\mathbf{x}_i),\varphi(\mathbf{x}_j)\rangle_{\mathcal{H}} = \varphi(\mathbf{x}_i)^{\mathrm{T}}\varphi(\mathbf{x}_j) = \kappa(\mathbf{x}_i,\mathbf{x}_j)$, where κ is a kernel function. Similar to LSDT, the objective function of NLSDT can be formulated as

$$\mathcal{J}(\boldsymbol{\mathcal{P}},\mathbf{Z},\mathbf{X}_T,\mathbf{X}) = \|\mathbf{Z}\|_1 + \lambda_1\|\boldsymbol{\mathcal{P}}\varphi(\mathbf{X}_T) - \boldsymbol{\mathcal{P}}\varphi(\mathbf{X})\mathbf{Z}\|_{\mathrm{F}}^2 + \lambda_2\|\varphi(\mathbf{X}) - \boldsymbol{\mathcal{P}}^{\mathrm{T}}\boldsymbol{\mathcal{P}}\varphi(\mathbf{X})\|_{\mathrm{F}}^2 \tag{12}$$

Based on Proposition 1, the optimal mapping $\boldsymbol{\mathcal{P}}^*$ can be represented as $\boldsymbol{\mathcal{P}}^* = \mathbf{\Phi}^{\mathrm{T}}\varphi(\mathbf{X})^{\mathrm{T}}$. The objection (12) becomes

$$\begin{aligned}\mathcal{J}(\boldsymbol{\mathcal{P}},\mathbf{Z},\mathbf{X}_T,\mathbf{X}) &= \|\mathbf{Z}\|_1 + \lambda_1\left\|\mathbf{\Phi}^{\mathrm{T}}\varphi(\mathbf{X})^{\mathrm{T}}\varphi(\mathbf{X}_T) - \mathbf{\Phi}^{\mathrm{T}}\varphi(\mathbf{X})^{\mathrm{T}}\varphi(\mathbf{X})\mathbf{Z}\right\|_{\mathrm{F}}^2 \\ &\quad + \lambda_2\left\|\varphi(\mathbf{X}) - \varphi(\mathbf{X})\mathbf{\Phi\Phi}^{\mathrm{T}}\varphi(\mathbf{X})^{\mathrm{T}}\varphi(\mathbf{X})\right\|_{\mathrm{F}}^2 \\ &= \|\mathbf{Z}\|_1 + \lambda_1\left\|\mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}_T - \mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}\mathbf{Z}\right\|_{\mathrm{F}}^2 + \lambda_2\left\|\varphi(\mathbf{X}) - \varphi(\mathbf{X})\mathbf{\Phi\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}\right\|_{\mathrm{F}}^2\end{aligned} \tag{13}$$

where $\boldsymbol{\mathcal{K}}_T = \varphi(\mathbf{X})^{\mathrm{T}}\varphi(\mathbf{X}_T)$ and $\boldsymbol{\mathcal{K}} = \varphi(\mathbf{X})^{\mathrm{T}}\varphi(\mathbf{X})$ denote the kernel Gram matrix. Therefore, the minimization problem of NLSDT can be written as

$$\min_{\mathbf{Z},\mathbf{\Phi}}\|\mathbf{Z}\|_1 + \lambda_1\|\mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}_T - \mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}\mathbf{Z}\|_{\mathrm{F}}^2 + \lambda_2 Tr\left((\mathbf{I} - \mathbf{\Phi\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}})^{\mathrm{T}}\boldsymbol{\mathcal{K}}(\mathbf{I} - \mathbf{\Phi\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}})\right) \tag{14}$$

$$\text{s.t. } \mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}\mathbf{\Phi} = \mathbf{I}, \mathbf{1}_{N_S+N_T}^{\mathrm{T}}\mathbf{Z} = \mathbf{1}_{N_T}^{\mathrm{T}}, Z_{N_S+i,i} = 0\ \forall i = 1,\cdots,N_T$$

```
Algorithm 3. The proposed LSDT
Input: $\mathbf{X}_S \in \Re^{D\times N_S}, \mathbf{X}_T \in \Re^{D\times N_T}, \mathbf{X} \in \Re^{D\times N}$, $\lambda_1$, $\lambda_2$
Procedure:
  1. Compute $\mathbf{K}_T := \mathbf{X}^{\mathrm{T}}\mathbf{X}_T$ and $\mathbf{K} := \mathbf{X}^{\mathrm{T}}\mathbf{X}$
  2. Perform eigenvalue decomposition $\mathbf{K} = \mathbf{VSV}^{\mathrm{T}}$
  3. Initialize $\mathbf{\Phi} := \mathbf{V}(:, \mathcal{V})$, where $\mathcal{V}$ is index of the $d$ largest eigenvalues
  4. while not converge do
  4.1. Step 1: fix $\mathbf{\Phi}$, and update $\mathbf{Z}$ in Problem (10) using Algorithm 1 based on
       ADMM
  4.2. Step 2: Fix $\mathbf{Z}$, and update $\mathbf{\Phi}$ in Problem (11) using Algorithm 2 based on
       Proposition 2 with eigenvalue decomposition
  4.3. Check convergence
  end while
Output: $\mathbf{\Phi}$ and $\mathbf{Z}$
```

```
Algorithm 4. The proposed NLSDT
Input: $\mathbf{X}_S \in \Re^{D\times N_S}, \mathbf{X}_T \in \Re^{D\times N_T}, \mathbf{X} \in \Re^{D\times N}$, $\lambda_1$, $\lambda_2$
Procedure:
  1. Compute $\boldsymbol{\mathcal{K}}_T := \varphi(\mathbf{X})^{\mathrm{T}}\varphi(\mathbf{X}_T)$ and $\boldsymbol{\mathcal{K}} := \varphi(\mathbf{X})^{\mathrm{T}}\varphi(\mathbf{X})$
  2. Perform Eigen-value decomposition $\boldsymbol{\mathcal{K}} = \mathbf{VSV}^{\mathrm{T}}$
  3. Initialize $\mathbf{\Phi} := \mathbf{V}(:, \mathcal{V})$, where $\mathcal{V}$ is index of the $d$ largest Eigen-values
  4. while not converge do
  4.1. Step 1: fix $\mathbf{\Phi}$, and update $\mathbf{Z}$ in (14) using Algorithm 1 by solving
       $\min_{\mathbf{Z}} \|\mathbf{Z}\|_1 + \lambda_1\|\mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}_T - \mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}\mathbf{Z}\|_{\mathrm{F}}^2$, s.t. $\mathbf{1}_{N_S+N_T}^{\mathrm{T}}\mathbf{Z} = \mathbf{1}_{N_T}^{\mathrm{T}}$
  4.2. Step 2: Fix $\mathbf{Z}$, and update $\mathbf{\Phi}$ in (14) using Algorithm 2 by solving
       $\min_{\mathbf{\Phi}} \lambda_1\|\mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}_T - \mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}\mathbf{Z}\|_{\mathrm{F}}^2 + \lambda_2 Tr\left((\mathbf{I} - \mathbf{\Phi\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}})^{\mathrm{T}}\boldsymbol{\mathcal{K}}(\mathbf{I} -$
       $\mathbf{\Phi\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}})\right)$, s.t. $\mathbf{\Phi}^{\mathrm{T}}\boldsymbol{\mathcal{K}}\mathbf{\Phi} = \mathbf{I}$
  4.3. Check convergence
  end while
Output: $\mathbf{\Phi}$ and $\mathbf{Z}$.
```

B. Optimization Algorithm

The optimization algorithm of NLSDT is similar with LSDT shown in Algorithm 1, in terms of Proposition 1 and Proposition 2. From the models (9) and (14), we can observe that LSDT is in fact a special case of NLSDT when a linear kernel function is used to compute $\boldsymbol{\mathcal{K}}_T$ and $\boldsymbol{\mathcal{K}}$. In NLSDT, Gaussian RBF function, sigmoid function, etc. can be used as kernel function. The NLSDT is illustrated in Algorithm 4.

C. Classification

With the case of NLSDT, the classification scheme in this paper consists of the following steps:

- Compute the latent subspace embedding $\mathbf{M}_S$ of source data $\mathbf{X}_S$ using the projection $\boldsymbol{\mathcal{P}}^*$, as $\mathbf{M}_S = \boldsymbol{\mathcal{P}}^*\varphi(\mathbf{X}_S)$.
- Compute the latent subspace embedding $\mathbf{M}_{Tl}$ of the *labeled* target training data $\mathbf{X}_{Tl}$ using the learned projection $\boldsymbol{\mathcal{P}}^*$ and sparse reconstruction $\mathbf{Z}$, as $\mathbf{M}_{Tl} = \boldsymbol{\mathcal{P}}^*\varphi(\mathbf{X})\mathbf{Z}$.
- Compute the latent subspace embedding $\mathbf{M}_{Tu}$ of these *unlabeled* target test data $\mathbf{X}_{Tu}$ as $\mathbf{M}_{Tu} = \boldsymbol{\mathcal{P}}^*\varphi(\mathbf{X}_{Tu})$.
- Train a classifier $\mathbf{W}$ using ℓ_2-norm regularized least square method on the *labeled* training data $[\mathbf{M}_S, \mathbf{M}_{Tl}]$ and label matrix $\mathbf{Y} = \left[\mathbf{Y}_S^{\mathrm{T}}, \mathbf{Y}_{Tl}^{\mathrm{T}}\right]^{\mathrm{T}}$, where $[\mathbf{Y}]_{i,j} = 1$ if the class j is assigned to the i-th sample, and -1 otherwise.
- The decision labels of *unlabeled* target test data are obtained by computing $\mathbf{M}_{Tu}^{\mathrm{T}}\mathbf{W}$.

V. Experiments

A. Synthetic Data

In this section, we use the generated toy data for latent subspace alignment by our method. The 3-dimensional source, few labeled target data and unlabeled target data with two classes generated by Gaussian distributions of different means and covariance matrices are shown in Fig. 3 (left). Each class in source domain contains 50 samples and it is easy to find a decision boundary of the two classes in source domain. In target domain, there are 5 labeled samples and 50 unlabeled samples for each class. From the figure, it is clearly observed that: 1) the data points of the same class between source and target domain have very different distribution; 2) the classification hyper-plane of source domain does not fit the decision boundary of target domain. Therefore, how to determine one robust decision boundary becomes very challenging.

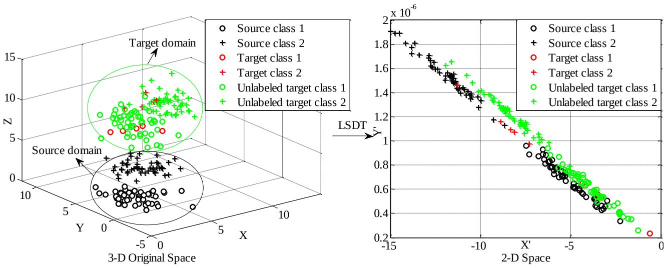


Fig. 3. The 3-D illustration of synthetic data (left) and 2-D illustration after subspace alignment (right)

TABLE I

3DA and 4DA Benchmark Datasets For Visual Domain Adaptation in Experiments

Dataset	Domain	#class	#dimension	#samples	n_s/c	n_t/c
3DA	Amazon	31	800	2813	20	3
	DSLR	31	800	498	8	3
	Webcam	31	800	795	8	3
4DA	Amazon	10	800	958	20	3
	DSLR	10	800	157	8	3
	Webcam	10	800	295	8	3
	Caltech	10	800	1123	8	3

The proposed LSDT aims to find a latent space with domain adaptation, such that both domains can have similar distribution and better separable ability in the latent space. By using the proposed LSDT method, the source data and target data in the 2-D subspace after projection and reconstruction can be seen in Fig. 3 (right). We can observe that the subspace mismatch between source data and target data is reduced after LSDT, and the decision boundary between the two classes is clear and easily to find with a general classifier. The toy data primarily demonstrates the effectiveness of our method in latent subspace alignment for representation based adaptation.

B. Object Recognition

In this section, cross-domain object recognition is discussed.

- **Experimental setup**

In experiments, we test our methods in two visual benchmark datasets: 3DA and 4DA of objects, which are widely used for domain adaptation. Besides, the deep features of 4DA datasets based on convolutional neural network (CNN) [38] are also exploited for object recognition. Specifically, the 3DA, 4DA and 4DA-CNN datasets and features are illustrated as follows.

- *3DA: Amazon*, *DSLR* and *Webcam* domain adaptation [9]

In the *3DA* dataset, each domain contains 31 object classes, such as back-pack, keyboard, earphone, etc. By following the setting in [9], if Amazon is experimented as source domain, 20 samples per class are selected for training, and 8 samples are selected if DSLR or Webcam is source domain. For target domain, 3 training samples per class are selected and the rest

TABLE II
RECOGNITION ACCURACY (%) OF SINGLE-SOURCE AND MULTI-SOURCE DOMAIN ADAPTATION IN 3DA SETTING

Domains		Compared methods							Our method	
Source	Target	ASVM [12]	GFK [10]	SGF [8]	SA [41]	RDALR [2]	LTSL-PCA[1]	LTSL-LDA [1]	LSDT	NLSDT
Amazon	Webcam	42.2±0.9	46.4±0.5	45.1±0.6	48.4±0.6	50.7±0.8	49.8±0.4	**53.5±0.4**	50.0±1.3	**56.3±0.7**
DSLR	Webcam	33.0±0.8	61.3±0.4	61.4±0.4	61.8±0.9	36.9±1.9	62.4±0.3	54.4±0.4	**69.4±0.7**	**69.9±0.3**
Webcam	DSLR	26.0±0.7	66.3±0.4	63.4±0.5	65.7±0.5	32.9±1.2	63.9±0.3	59.1±0.5	**72.6±0.9**	**74.6±0.5**
Amazon+DSLR	Webcam	30.4±0.6	34.3±0.6	31.0±1.6	54.4±0.9	36.9±1.1	55.3±0.3	30.2±0.5	**69.0±0.8**	**66.1±0.7**
Amazon+Webcam	DSLR	25.3±1.1	52.0±0.8	25.0±0.4	37.5±1.0	31.2±1.3	57.7±0.4	43.0±0.3	**67.5±1.8**	**65.7±0.9**
DSLR+Webcam	Amazon	17.3±0.9	21.7±0.5	15.0±0.4	16.5±0.4	20.9±0.9	20.0±0.2	17.1±0.3	**22.0±0.1**	**23.2±0.6**

TABLE III
RECOGNITION ACCURACY (%) OF DIFFERENT DOMAIN ADAPTATION OVER 10 OBJECT CATEGORIES IN 4DA SETTING

Task	Compared methods											Our methods	
	Na ïve Comb	ARC-t [15]	SGF [8]	GFK [10]	DAM [18]	MMDT [14]	Symm [19]	SA [41]	DIP [42]	LTSL-PCA [1]	LTSL-LDA [1]	LSDT	NLSDT
A→D	55.9±0.8	50.2±0.7	46.9±1.1	50.9±0.9	57.8±0.8	56.7±1.3	47.9±1.4	55.1	42.8	50.4±0.9	**59.1±0.7**	52.9±0.8	**60.7±0.8**
C→D	55.8±0.9	50.6±0.8	50.2±0.8	55.0±0.9	58.5±0.7	56.5±0.9	48.6±1.1	56.6	49.0	49.5±0.8	**59.6±0.6**	56.0±0.8	**62.2±0.9**
W→D	55.1±0.8	71.3±0.8	78.6±0.4	75.0±0.7	68.2±0.5	67.0±1.1	69.8±1.0	82.3	**86.4**	**82.6±0.5**	**82.6±0.5**	75.7±0.9	76.5±0.6
A→C	32.0±0.8	37.0±0.4	37.5±0.4	39.6±0.4	39.6±0.1	36.4±0.8	39.1±0.5	38.4	**43.3**	41.5±0.3	39.8±0.3	42.2±0.3	**46.8±0.4**
W→C	30.4±0.7	31.9±0.5	32.9±0.7	32.8±0.7	**37.1±0.1**	32.2±0.8	34.0±0.5	34.1	37.0	36.7±0.3	**38.5±0.3**	36.9±0.3	**40.3±0.5**
D→C	31.7±0.6	33.5±0.4	32.9±0.4	33.9±0.6	36.5±0.1	34.1±0.8	34.9±0.4	35.8	**39.0**	36.2±0.3	36.7±0.4	37.6±0.4	**41.0±0.5**
D→A	45.7±0.9	42.5±0.5	44.9±0.7	46.2±0.6	46.0±0.2	46.9±1.0	42.7±0.5	45.8	40.5	45.7±0.3	**47.4±0.5**	46.6±0.5	**56.1±0.6**
W→A	45.6±0.7	43.4±0.5	43.0±0.7	46.2±0.7	45.6±0.1	47.7±0.9	43.7±0.7	44.8	42.5	41.9±0.3	**47.8±0.4**	46.6±0.5	**54.5±0.6**
C→A	45.3±0.9	44.1±0.6	42.0±0.5	46.1±0.6	**51.9±0.2**	49.4±0.8	43.8±0.6	45.3	50.0	49.3±0.4	**50.4±0.5**	47.7±0.5	**54.2±0.6**
C→W	60.3±1.0	55.9±1.0	54.2±0.9	57.0±0.9	**63.8±0.5**	**63.8±1.1**	50.5±1.6	**60.7**	47.6	50.4±0.8	59.5±0.8	57.6±0.9	**64.3±1.2**
D→W	62.1±0.8	78.3±0.5	78.6±0.4	80.2±0.4	76.4±0.3	74.1±0.8	78.4±0.9	**84.8**	**86.7**	81.0±0.5	78.3±0.4	83.1±0.4	83.5±0.3
A→W	62.4±0.9	55.7±0.9	54.2±0.8	56.9±1.0	61.2±0.4	**64.6±1.2**	51.0±1.4	**60.3**	46.7	52.3±0.8	59.5±1.1	57.2±0.9	**65.2±1.0**

data in the target domain is used for testing. The detail of *3DA* dataset is summarized in Table I.

- *4DA: Amazon*, *DSLR*, *Webcam* and *Caltech 256* [10]

For *4DA* dataset, four domains are included, where each domain contains 10 common object classes rather than 31 selected from 3DA dataset and an extra Caltech 256 dataset [11]. In experiments, we follow the configuration in [10] where 20 samples per class are selected from Amazon, and 8 samples per class are randomly selected from DSLR, Webcam and Caltech if they are source domains, while 3 samples per category are selected if they are target domains, and the rest data in target domain is used for testing. The detail of *4DA* dataset is also summarized in Table I. Note that, the 800-bin SURF features provided in [9, 10] for each domain are used.

- *4DA-CNN: Amazon*, *DSLR*, *Webcam* and *Caltech 256* domain adaptation [10, 39]

For the *4DA-CNN* setting, 8 layers with 5 convolutional layers and 3 fully connected layers of CNN were trained on ImageNet in [38]. The well-trained CNN structure and parameters are used by taking the 4DA dataset as input of CNN [39]. The outputs of the 6th and 7th layer (i.e. DeCAF) are used. The feature dimension after CNN is 4096. More details of the architecture and training protocol can be referred to [38, 39].

- *Parameter Setting*

For LSDT method, the trade-off coefficients λ_1 and λ_2 are fixed to be 1 in experiments. For NLSDT, the Gaussian function $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2\right)$ is used, and the kernel parameter σ is tuned for the best result. The ℓ_2-norm regularized least square method is used for classifier training.

- **3DA experiment**

We strictly follow the experimental configuration by Saenko *et al.* [9]. 20 random splits of training data in source and target domain are implemented and the mean accuracies over 31 categories are reported. The experiments are employed in single source domain and multiple source domains adaptation, respectively. In this experiment, we compare with five methods including ASVM [12], GFK [10], SGF [8], SA [41], RDALR [2], LTSL-PCA [1] and LTSL-LDA [1]. The experimental results of single source domain and multiple source domains adaptation are shown in Table II.

From the results, we can observe that LSDT with nonlinear kernel function performs much better results than other methods for single source domain adaptation. For multiple source domain adaptation, both LSDT and NLSDT outperform other methods. However, NLSDT is a little weak compared to the linear method. Note that partial results of other methods are quoted from [1, 2].

Additionally, in Table II, the LTSL-PCA is better than LTSL-LDA a. Note that LTSL outperforms RDALR method with a large margin which shows that the subspace learning is beneficial to domain transfer. Therefore, in the subsequent experiments, LTSL as *low-rank* based subspace adaptation is compared, instead of RDALR.

- **4DA experiment**

In this experiment, we strictly follow the experimental setting by Gong *et al.* [10]. There are four domains, and therefore 12 combinations of each two domains are discussed. 20 random splits of training data in source and target domain are used for all methods, and the mean classification accuracies over 10 object categories are reported in Table III. Note that A: Amazon, D: DSLR, W: Webcam, C: Caltech 256. We have compared to existing methods including Na ïveComb, ARC-t [15], sampling geodesic flow (SGF) [8], geodesic flow kernel (GFK) [10], domain adaptation machine (DAM) [18], max-margin domain transforms (MMDT) [14], Symm [19], SA

TABLE IV
RECOGNITION ACCURACY (%) OVER 10 OBJECT CATEGORIES IN 4DA-CNN SETTING WITH DEEP FEATURE REPRESENTATION

Method	Layer	A→D	C→D	W→D	A→C	W→C	D→C	D→A	W→A	C→A	C→W	D→W	A→W
SourceOnly	f_6	80.8±0.8	76.6±2.2	96.1±0.4	79.3±0.3	59.5±0.9	67.3±1.2	77.0±1.0	66.8±1.0	85.8±0.4	67.5±1.6	95.4±0.6	70.5±0.9
	f_7	81.3±0.7	77.6±1.1	96.2±0.6	79.3±0.3	68.1±0.6	74.3±0.6	81.8±0.5	73.4±0.7	86.5±0.5	67.8±1.8	95.1±0.8	71.6±0.6
Na ïveComb	f_6	94.5±0.4	92.9±0.8	99.1±0.2	84.0±0.3	81.7±0.5	83.0±0.3	90.5±0.2	90.1±0.2	89.9±0.2	91.6±0.8	97.9±0.3	90.4±0.8
	f_7	94.1±0.8	92.8±0.7	98.9±0.2	83.4±0.4	81.2±0.4	82.7±0.4	90.9±0.3	90.6±0.2	90.3±0.2	90.6±0.8	98.0±0.2	91.1±0.8
SGF [8]	f_6	90.5±0.8	93.1±1.2	97.7±0.4	77.1±0.8	74.1±0.8	75.9±1.0	88.0±0.8	87.2±0.5	88.5±0.4	89.4±0.9	96.8±0.4	87.2±0.9
	f_7	92.0±1.3	92.4±1.1	97.6±0.5	77.4±0.7	76.8±0.7	78.2±0.7	88.0±0.5	86.8±0.7	89.3±0.4	87.8±0.8	95.7±0.8	88.1±0.8
GFK [10]	f_6	92.6±0.7	92.0±1.2	97.8±0.5	78.9±1.1	77.5±0.8	78.8±0.8	88.9±0.3	86.2±0.8	87.5±0.3	87.7±0.8	97.0±0.8	89.5±0.8
	f_7	94.3±0.7	91.9±0.8	98.5±0.3	79.1±0.7	76.1±0.7	77.5±0.8	90.1±0.4	85.6±0.5	88.4±0.4	86.4±0.7	96.5±0.3	88.6±0.8
SA [41]	f_6	94.2±0.5	93.0±1.0	98.6±0.5	83.1±0.7	81.1±0.5	82.4±0.7	90.4±0.4	89.8±0.4	89.5±0.4	91.2±0.9	97.5±0.7	90.3±1.2
	f_7	92.8±1.0	92.1±0.9	98.5±0.3	83.3±0.2	81.0±0.6	82.9±0.7	90.7±0.5	90.9±0.4	89.9±0.5	89.0±1.1	97.5±0.4	87.8±1.4
LTSL-PCA [1]	f_6	94.6±0.6	93.4±0.6	99.2±0.2	85.5±0.3	82.0±0.5	84.7±0.5	91.2±0.2	89.5±0.2	91.3±0.2	90.2±0.8	97.0±0.5	89.4±1.2
	f_7	95.7±0.5	94.6±0.8	98.4±0.2	86.0±0.2	83.5±0.4	85.4±0.4	92.3±0.2	91.5±0.2	92.4±0.2	90.9±0.9	96.5±0.2	91.2±1.1
LTSL-LDA [1]	f_6	95.5±0.3	93.6±0.5	99.1±0.2	85.3±0.2	82.3±0.4	84.4±0.2	91.1±0.2	90.6±0.2	90.4±0.1	91.8±0.7	98.2±0.3	92.2±0.4
	f_7	94.5±0.5	93.5±0.8	98.8±0.2	85.4±0.1	82.6±0.3	84.8±0.2	91.9±0.2	91.0±0.2	90.9±0.1	90.8±0.7	97.8±0.3	91.5±0.5
LSDT	f_6	**96.4±0.4**	**95.4±0.5**	**99.4±0.1**	85.9±0.2	83.1±0.3	85.2±0.2	92.2±0.2	91.0±0.2	92.1±0.1	**93.3±0.8**	**98.7±0.2**	92.1±0.8
	f_7	96.0±0.4	94.6±0.5	99.3±0.1	**87.0±0.2**	**84.2±0.3**	**86.2±0.2**	**92.5±0.2**	**91.7±0.2**	**92.5±0.1**	**93.5±0.8**	98.3±0.2	92.9±0.8
NLSDT	f_6	**96.4±0.4**	**95.7±0.5**	**99.5±0.1**	85.8±0.2	83.3±0.3	85.3±0.2	92.3±0.2	91.1±0.2	91.9±0.1	92.9±0.7	**98.6±0.2**	**94.2±0.4**
	f_7	96.0±0.4	94.4±0.8	99.4±0.2	**86.9±0.2**	**84.3±0.3**	**86.2±0.2**	**92.5±0.2**	**91.9±0.2**	**92.3±0.1**	93.2±0.8	98.1±0.3	**94.1±0.4**

[41], DIP [42] and LTSL [1]. From the results, we can observe that NLSDT performs much better than state-of-the-art LTSL results and is also superior to other methods. Particularly, the average performance over 12 different tasks of our NLSDT is about 5% improvement compared to LTSL. We can also see that for LTSL, LDA is better than PCA for subspace learning. Additionally, the results also demonstrate that nonlinear method is effective for domain adaptation, since nonlinear shift may occur in data acquisition.

- **4DA-CNN experiment**

The experimental setting is the same as *4DA* setting, but with CNN features. The comparison results with state-of-the-art representation based domain adaptation methods such as SGF [8], GFK [10], SA [41], LTSL-PCA [1] and LTSL-LDA [1], are reported in Table IV. Note that *SourceOnly* denotes the results trained by SVM on the source data, *Na ïveComb* denotes the baseline method learned by SVM on the combined source and target training data. From Table IV, we observe that: 1) the total classification performance is well improved by using deep feature representation, for example, the classification accuracy increases from 83.5% to 98.7% for "D→W" setting by using our NLSDT method, which show that the deep feature representation can effectively remove the domain shift or bias; 2) LSDT and NLSDT have similar performance on deep features, which implies the linearly separable ability of the high-level deep representation; 3) the proposed methods still outperform other methods; 4) the output features of the 6th and 7th layer have comparative performance in object recognition.

C. *Consumer & YouTube Video Event Recognition*

In this experiment, the dataset used for video event recognition is the YouTube videos & Consumer videos developed in [16], in which part of consumer videos were from Kodak Consumer video benchmark dataset [27] and part from real users. Considering that in real applications the labeled samples of consumer videos are usually fewer than the labeled web videos, the web videos of low-resolution from YouTube website are used as source data, while the consumer videos of high-resolution are used as target data in experiments.

By following [16], six visual events including "birthday", "picnic", "parade", "show", "sports", and "wedding" are included. The total number of YouTube videos and Consumer videos is 906 and 195, respectively. For source domain, all 906 YouTube web videos are used as labeled source data. For target domain, we randomly selected m (m=1, 3, 5, 7, 10) consumer videos per event as the labeled target training data, and the remaining consumer videos are used as unlabeled data for evaluation. We sample the labeled target training videos 5 times, the means and standard deviations of classification accuracies are reported.

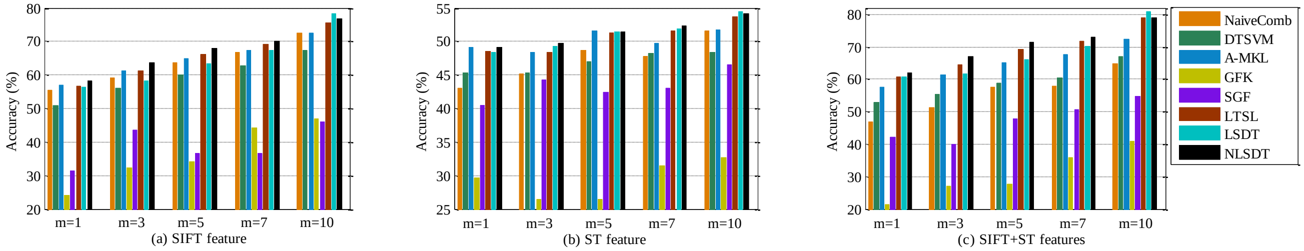
As described in [16], two types of features, ST feature [28] and SIFT feature [29] are used. For ST feature, 72D HOG and 90D HOF features are concatenated as a 162D vector. For each frame (65 frames per video), 128D SIFT features are extracted from salient regions detected by DoG interest point detector [30]. Finally, the visual *vocabularies* via k-means are built for feature clustering. The features can be obtained from [16].

In experiment, we have compared our proposed method with two classifier based transfer learning methods such as A-MKL [16] and DTSVM (DTMKL) [17] which report the state-of-the-art results on this dataset, and three representation based domain adaptation methods such as GFK [10], SGF [8] and LTSL [1] coupled with PCA and LDA. The basic idea of A-MKL method is to learn the target classifier $f^T(\mathbf{x})$ with the optimal combination $\sum_{p=1}^{P} \beta_p f_p^S(\mathbf{x})$ of P source classifiers plus a learned perturbation $\Delta f(\mathbf{x}) = \sum_{m=1}^{M} d_m \mathbf{w}_m \varphi_m(\mathbf{x}) + b$ based on multiple kernels. The basic idea of DTSVM (DTMKL) tends to learn target decision function $f^T(\mathbf{x}) = \sum_{m=1}^{M} d_m \mathbf{w}_m \varphi_m(\mathbf{x}) + b$ without considering the optimal combination of pre-learned source classifiers involved in A-MKL. We also compared the baseline method (i.e. Na ïveComb) trained by SVM.

We have studied the recognition performance by leveraging different number m (m=1, 3, 5, 7, 10) of labeled videos per event from consumer videos (target domain). The recognition accuracies over 6 visual events based on three types of low-level features are reported in Table V. From the results, we can find that the proposed LSDT method with nonlinear Gaussian kernel outperforms other methods. Fig. 4 describes the recognition accuracy of all methods with the increasing

Table V
CLASSIFICATION ACCURACY (%) OVER 6 VISUAL EVENTS WITH DIFFERENT NUMBER OF LABELED TARGET VIDEOS PER EVENT

m	Feature	Compared methods								Our methods	
		Na ïve Comb	DTSVM [16]	A-MKL [17]	GFK [10]	SGF [8]	SA [41]	LTSL-PCA [1]	LTSL-LDA [1]	LSDT	NLSDT
1	SIFT	55.4±0.5	50.9±0.9	57.0±0.8	24.0±0.4	31.4±3.6	43.5±1.4	**57.8±0.8**	55.7±0.5	56.3±0.2	**58.3±0.1**
	ST	43.1±0.3	45.3±0.2	**49.2±0.4**	29.7±0.9	40.5±0.4	45.5±0.6	48.2±0.3	47.5±0.1	48.4±0.4	**49.2±0.3**
	SIFT+ST	46.8±0.6	52.8±0.4	57.7±0.8	21.4±1.0	42.2±2.1	46.0±1.1	59.6±1.3	60.7±0.7	**60.8±0.8**	**62.0±0.6**
3	SIFT	59.1±1.2	56.0±0.5	61.4±1.3	32.3±1.6	43.6±1.6	50.3±2.5	**62.3±0.9**	61.3±0.4	58.2±0.6	**63.8±1.1**
	ST	45.2±0.9	45.4±0.7	48.3±0.4	26.4±0.7	44.3±0.5	44.3±0.5	49.3±0.1	48.4±0.3	**49.3±0.3**	**49.7±0.2**
	SIFT+ST	51.2±1.6	55.4±0.0	61.5±1.0	27.0±1.6	40.1±4.2	46.4±0.6	**62.8±1.0**	60.5±0.6	61.8±1.2	**67.1±0.6**
5	SIFT	63.8±2.0	60.0±1.6	**65.0±2.2**	34.2±0.9	36.5±2.5	60.4±2.9	64.7±1.0	63.0±1.8	63.3±1.3	**67.9±1.8**
	ST	48.7±1.4	47.0±0.6	**51.5±0.8**	26.4±0.9	42.4±0.0	45.9±0.3	50.8±0.5	51.3±0.3	51.4±0.2	**51.5±0.3**
	SIFT+ST	57.5±1.8	58.8±0.6	65.3±2.2	27.6±0.9	47.9±2.4	53.2±2.8	62.3±0.7	64.1±2.1	**66.1±1.3**	**71.4±2.0**
7	SIFT	66.9±2.3	62.9±1.9	67.3±2.3	44.2±1.0	36.5±3.7	54.2±1.8	65.7±1.4	67.3±2.3	**67.3±1.7**	**70.0±2.1**
	ST	47.8±1.3	48.2±1.2	49.7±1.0	31.4±0.8	43.1±0.1	45.8±0.6	51.6±0.4	51.6±0.3	**51.9±0.4**	**52.3±0.4**
	SIFT+ST	57.9±1.5	60.5±1.2	67.7±2.4	36.0±0.9	50.8±2.8	53.2±2.9	68.6±1.8	66.9±2.4	**70.3±1.5**	**73.0±2.4**
10	SIFT	72.4±2.1	67.3±2.3	72.4±2.2	46.9±1.6	46.0±1.3	64.2±2.8	72.9±2.2	73.5±2.1	**78.3±1.4**	**76.7±2.3**
	ST	51.5±0.8	48.3±1.5	51.7±1.3	32.7±1.1	46.6±0.5	48.9±0.6	53.4±0.7	52.7±0.6	**54.4±0.6**	**54.2±0.8**
	SIFT+ST	65.0±0.9	67.2±1.7	72.4±2.3	41.0±1.1	54.7±2.0	56.5±2.3	69.6±1.0	75.1±1.9	**80.9±1.7**	**79.0±1.5**



(a) SIFT feature (b) ST feature (c) SIFT+ST features

Fig. 4. Recognition accuracy with different number of labeled videos per-event selected from target domain

number *m* of labeled videos per event. From the plots with different features, we can observe that the proposed LSDT and NLSDT methods still perform the best results.

D. CMU Multi-PIE Data

The CMU Multi-PIE face dataset [23] is a comprehensive face dataset of 337 subjects, in which the images are captured across 15 poses, 20 illuminations, 6 expressions and 4 different sessions. For our purpose, we select the first 60 subjects from session 1 and session 2 in experiments. Session 1 contains 7 images per subject with 7 poses under neutral expression, while session 2 was prepared with the same poses as session 1 but under smile expression. The example images of one subject in session 1 and session 2 are illustrated in Fig.5. In this experiment, four experimental configurations are as follows.

- *Session 1*: one frontal face in red Rectangle and one 60° posed face in blue per subject are used as source and target training data, respectively. The remaining faces are probe faces.
- *Session 2*: the same configuration as session 1 is conducted.
- *Session 1+2*: Two frontal faces and two faces with extreme 60° pose from both sessions are used as training data. The remaining faces are used as probe faces.
- *Cross session*: The faces per subject in session 1 with neural expression are taken as source domain, while the faces per subject in session 2 with smile expression are taken as target domain. This is to adapt the change of expression.

Pose alignment is challenging due to the highly non-linear changes induced by 3-D rotation of a face. Fig.6 illustrates the pose alignment process under Session 2 with smile expression by the proposed NLSDT, where the frontal faces per subject in red Rectangle are used as source data, and the faces with 60° poses in the blue Rectangle are used as target data for each session. From Fig. 6, we can observe that the target face under pose is well aligned with residual (noise) removed.

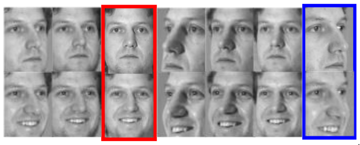


Fig. 5. Example images of one subject. Session 1 (the 1st row with neutral expression) and Session 2 (the 2nd row with smile expression)

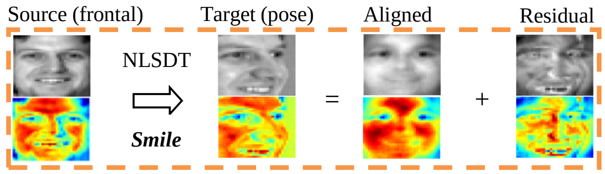


Fig. 6. Pose alignment of Session 2 by the proposed NLSDT method.

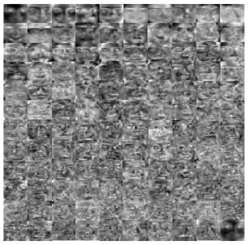


Fig. 7. Examples of the learned basis transformation **P** by NLSDT under Session 2. Each subplot represents a row of **P**.

The best face recognition rates under the four experimental configurations by using different methods are shown in Table VI. From the results, we can see that the proposed NLSDT significantly outperforms other state-of-the-art methods.

Table VI
COMPARISON WITH OTHER METHODS FOR FACE RECOGNITION ACROSS POSES

Domains			Compared methods							Our method	
Tasks	Source	Target	Na ïveComb	ASVM [12]	SGF [8]	GFK [10]	SA [41]	LTSL-PCA [1]	LTSL-LDA [1]	LSDT	NLSDT
Session 1	frontal	60° pose	61.0	57.0	53.7	56.0	51.3	55.7	56.0	59.7	**63.7**
Session 2	frontal	60° pose	62.7	62.7	55.0	58.7	62.7	58.7	60.7	63.3	**70.7**
Session 1+2	frontal	60° pose	60.2	60.1	53.8	56.3	61.7	57.8	60.7	61.7	**67.5**
Cross session	Session 1	Session 2	93.6	94.3	92.5	96.7	98.3	96.7	96.7	95.8	**99.4**

Table VII
HANDWRITTEN DIGITS RECOGNITION PERFORMANCE ACROSS DOMAINS

Domains		Compared methods							Our method	
Source	Target	Na ïve Comb	A-SVM [12]	SGF [8]	GFK [10]	SA [41]	LTSL-PCA [1]	LTSL-LDA [1]	LSDT	NLSDT
MNIST	USPS	78.8±0.5	78.3±0.6	79.2±0.9	82.6±0.8	78.8±0.8	83.2±0.9	78.4±0.7	79.3±0.8	**87.4±0.5**
SEMEION	USPS	83.6±0.3	76.8±0.4	77.5±0.9	82.7±0.6	82.5±0.5	83.6±0.3	83.4±0.3	84.7±0.4	**86.8±0.3**
MNIST	SEMEION	51.9±0.8	70.5±0.7	51.6±0.7	70.5±0.8	74.4±0.6	72.8±0.6	67.6±0.4	69.1±0.5	**79.6±0.8**
USPS	SEMEION	65.3±1.0	74.5±0.6	70.9±0.8	76.7±0.3	74.6±0.6	65.3±1.0	64.5±0.7	67.4±1.1	**81.9±0.7**
USPS	MNIST	71.7±1.0	73.2±0.8	71.1±0.7	74.9±0.9	72.9±0.7	71.7±1.0	71.2±1.0	70.5±1.4	**79.1±0.8**
SEMEION	MNIST	67.6±1.2	69.3±0.7	66.9±0.6	74.5±0.6	72.9±0.7	67.6±1.2	66.8±1.2	70.0±1.3	**75.4±0.8**

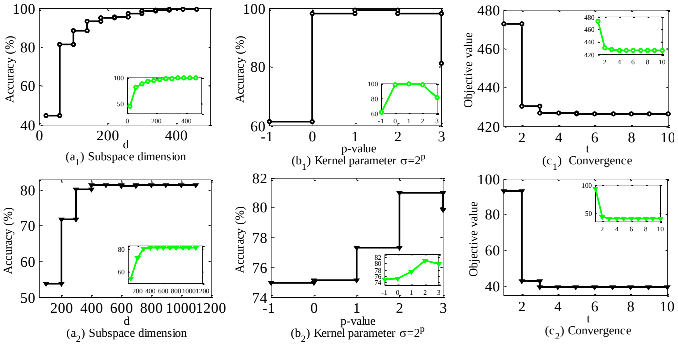
This demonstrates that linear subspace transfer may not work for nonlinear rotation. Fig.7 shows the learned basis **P** on Session 2. Each subplot corresponds to a row of **P**. The first 60 subplots denote the frontal source faces and the last 60 subplots show the target faces with 60° pose, from which we can observe that the target faces across poses can be aligned.

E. Handwritten Digits Data

In this section, three handwritten digits datasets including MNIST [24], USPS [25] and SEMEION [25] are used for cross-domain learning experiments, and the classification accuracies over 10 classes from digit 0 to digit 9 are reported for different tasks. The MNIST handwritten digits dataset has 70,000 instances with each image size of 28×28, the USPS dataset contains 9298 examples with each image size of 16×16, and 2593 images of size 16×16 are included in SEMEION dataset. For dimension consistency, the size of MNIST digit images is manually resized into 16×16.

In experiment, cross-domain tests are explored. Specifically, each dataset will be recognized to be source and target domain alternatively. Therefore, 6 combinations of cross-domain task are experimented. For the purpose of our experiments, we randomly select 100 samples per class from source domain for training and 10 samples per class from target domain for testing. 5 random splits are used and the mean accuracies via nearest neighbor classifier with the best parameter tuning are reported in Table VII, in which A-SVM [12], SGF [8], GFK [10], SA [41] and LTSL [1] are compared with our proposed NLSDT method with Gaussian kernel function used.

From the results, we can see that the proposed method outperforms other methods. The average improvement in accuracy is about 10% and 5% compared to the two methods, respectively. This demonstrates that the proposed NLSDT succeeds in dealing with highly nonlinear domain shift/bias.



(a_1) Subspace dimension (b_1) Kernel parameter σ=2^p (c_1) Convergence

(a_2) Subspace dimension (b_2) Kernel parameter σ=2^p (c_2) Convergence

Fig. 8. Performance with subspace dimension *d* (a), kernel parameter *σ* (b), and objective function with iteration *t* (c) on CMU Multi-PIE (a_1, b_1 and c_1) and Handwritten digits (a_2, b_2 and c_2). The *stair* curve in each subplot corresponds to the green curve.

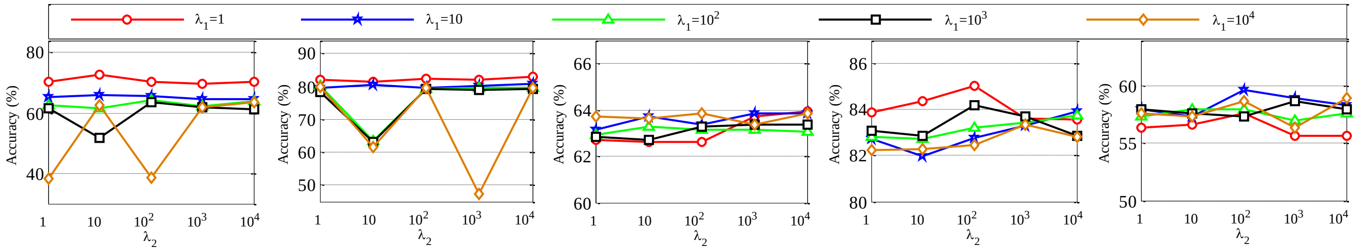
VI. DISCUSSION

A. Subspace dimension, Kernel parameter, and Convergence

This paper aims to learn a latent low dimensional subspace for representation based adaptation. For showing the performance with subspace dimension variation, we conduct the experiments on multi-PIE face data with Session 1 as source and Session 2 as target and handwritten digit data with USPS digits as source and SEMEION digits as target. Fig. 8 illustrates the performance of our method with increasing number of subspace dimension *d*, kernel parameter *σ* and iteration number *t*. From Fig. 8, it is clear that the proposed method can effectively learn a low-dimensional latent space and reduce the computational demand of sparse coding. Additionally, from the convergence curves of objective function, the model can converge to one minimum value after 3 iterations, which demonstrates the efficiency of optimization.

B. Parameter Sensitivity Analysis of LSDT

In the proposed LSDT model, there are two trade-off parameters λ_1 and λ_2 involved for model tuning. For insight of their sensitivity to the performance, we tune the two parameters from {1, 10, 100, 1000, 10000}, respectively, and report the accuracy on several datasets. Fig. 9 denotes the results *w.r.t.* different parameter values of λ_1 and λ_2. We see that the two parameters show relatively stable performance, except that for 3DA (a) and 4DA (b), the performance has a large variation when a large λ_1 is given. It is easy to obtain a relatively good performance by slightly tuning the model parameters.



(a) 3DA data (Dslr->Webcam) (b) 4DA data (D->W) (c) YouTube&Consumer Videos data (d) Handwritten digit data (Semeion->USPS) (e) CMU PIE data (session 1)

Fig. 9. Parameter sensitivity analysis on the considered datasets.

C. *Parameter Settings of Baseline Methods*

Throughout the paper, we have compared 13 methods including 4 adaptive classifier based methods such as ASVM, DAM, AMKL and DTSVM, 5 feature transformation based methods such as DIP, MMDT, Symm, ARC-t, and RDALR,, and 4 closely related methods such as GFK, SGF, SA and LTSL. We present the parameter discussion from three aspects:

- In the classifier based methods, SVM is an important tool in the models, such as ASVM and DAM. Therefore, the kernel parameter and penalty coefficient are the main parameters. For AMKL and DTSVM (also called DTMKL), multiple kernels are integrated for improving the domain transfer performance by minimizing the structural risk and the maximum mean discrepancy (MMD) between source and target domains. Therefore, the number of base kernels and kernel parameters play a key role in the optimal kernel function learning. In YouTube&Consumer video experiments, we have used the default parameters in the released codes and their reported results for comparisons.
- In the feature transformation based methods, DIP, MMDT, Symm and ARC-t tend to learn a transformation such that some similarity metric can be achieved with the maximized similarity or the minimized distance between the distribution of the transformed source and target data. RDALR tends to reconstruct the target data with source data by learning a low-rank matrix. Therefore, in these methods, one or two regularization parameters are referred during learning the transformation matrix. In the experiments, we have copied the accuracy result reported in their publications for comparisons.
- In the closely related methods, GFK, SGF, SA and LTSL methods have a common characteristic that the unsupervised subspace transfer is explored. Specifically, principal component analysis (PCA) is used for pre-learning the low-dimensional subspace, where the domain adaptation is implemented with different strategies. Therefore, the subspace dimension d is one key parameter for tuning in these subspace alignment based domain learning methods. Additionally, SGF associates with the number of PLS factors and LTSL refers to the trade-off parameter λ_2/λ_1 in (2). In this paper, the subspace dimension and trade-off parameters in these methods have been tuned, and the best results are reported for comparisons.

Table VIII
COMPARISON TO PCA AND LDA

Data	PCA-LSDT	LDA-LSDT	LSDT
Multi-PIE	56.7	45.3	59.7
3DA: A→W	52.5±0.8	38.3±0.8	69.4±0.7

Table IX
RECONSTRUCT THE TARGET DATA WITH SOURCE DATA ONLY (S) AND COMBINED SOURCE AND TARGET DATA (ST)

Data	LSDT-LRR(S)	LSDT-LRR(ST)	LSDT(S)	LSDT(ST)
Multi-PIE	52.0	57.3	55.0	59.7
3DA: D→W	59.5±0.3	63.0±0.2	59.7±0.3	69.4±0.7

D. *Pre-learn* **P** *using PCA and LDA in LSDT*

Following the pre-learning of subspace in LTSL, in this section, we discuss the joint learning of **P** and reconstruction **Z**, by comparing to PCA and LDA. The comparison results on multi-PIE and 3DA datasets are reported in Table VIII. The increments of recognition accuracy demonstrate the contribution of learning **P** simultaneously with **Z** in LSDT.

E. *Low-rank Constraint on* **Z** *in LSDT*

In LTSL, low-rank representation based reconstruction is used for subspace transfer. For demonstrating the effectiveness of LSDT based on SSC theory, we discuss the performance of LRR in LSDT in Table IX (LSDT-LRR vs. LSDT). The results demonstrate that LSDT based on SSC is significantly better than that of LRR -based.

F. *Reconstruct* $\mathbf{X}_T$ *using* $\mathbf{X}_S$ *Only in LSDT*

We have also discussed the performance comparison by reconstructing $\mathbf{X}_T$ using $\mathbf{X}$=[$\mathbf{X}_S$,$\mathbf{X}_T$] (ST) and $\mathbf{X}_S$ (S), respectively. The results in Table IX denote that the performance can be well improved by reconstructing the target data using both source and target data in domain transfer. Generally, in reconstruction based domain adaptation, when only a few number of source data is available, the target data can be leveraged for robust subspace transfer. It is worth noting that sparse coding requires sufficient data for obtaining an over-complete dictionary (i.e. $\mathbf{X}_S$). For domain adaptation, when the source data are insufficient, the assumption on over-complete dictionary may not hold. In this work, we adopt two strategies to avoid this issue. First, we consider the [$\mathbf{X}_S$,$\mathbf{X}_T$] as the dictionary for reconstruction to enlarge the dictionary size. Second, by introducing the low-dimensional projection **P**, we consider the reconstruction of $\mathbf{PX}_T$ by using the dictionary $\mathbf{P}$[$\mathbf{X}_S$,$\mathbf{X}_T$]. Therefore, even the dictionary $\mathbf{X}_S$ is not over-complete for coding $\mathbf{X}_T$, the dictionary $\mathbf{P}$[$\mathbf{X}_S$,$\mathbf{X}_T$] will be over-complete for coding $\mathbf{PX}_T$.

G. Justification of Motivations

The proposed LSDT is motivated by SSC theory, and aims at realizing unsupervised domain adaptation by exploiting sparse reconstruction between different domains in the latent subspace. The in-depth approach motivation of LSDT is as follows.

1)In general, the data from different domains lie in a union of multiple subspaces. For knowledge "transfer" but not naïve "transformation", the low-dimensional latent space of data should be found. Then, the "transfer" task can be effectively explored without overfitting. For finding such a latent space, we propose to learn a subspace projection **P**. In LTSL [1], the PCA or LDA is used to compute the **P** for subspace pursuit.

2)After obtaining the latent space via the **P**, the knowledge "transfer" is then implemented. In this paper, the "transfer" is realized via a reconstruction **Z**. In general, a good reconstruction is very important for robust domain adaptation. *First*, the outliers (noise) from source domain would be removed in transferring to the target domain; *Second*, fewer data from source domain should be used for reconstruction. For this reason, we propose to impose a sparse constraint on **Z**. The superiority is shown in Table IX.

3)From the above motivation 1) and 2), the reason why we learn **P** and **Z** is clear. Although the **P** can be pre-computed by existing subspace learning methods, it is sub-optimal and leads to the local optimum of **Z**. Therefore, we propose to learn the **P** and **Z** simultaneously by using an alternative strategy, such that a much better solution with stronger domain adaptability can be achieved. The performance comparison is demonstrated in Table VIII.

4)We aim to reconstruct the target data $\mathbf{P}\mathbf{X}_T$ by using $\mathbf{P}[\mathbf{X}_S,\mathbf{X}_T]$. The $\mathbf{X}_S$ part is used for knowledge adaptation and the $\mathbf{X}_T$ part is exploited for self-representation and outlier removing from the target data. When only a few source data is available, the the robustness can be improved by leveraging the target data in reconstruction. Note that the trivial solution of **Z** is avoided based on the SSC theory instead of LRR. The performance can be observed in Table IX.

5)The proposed NLSDT is an extension of LSDT, which is motivated by the highly nonlinear domain shift. By simply introducing kernel function into LSDT, the performance is greatly improved throughout the experiments.

VII. Conclusion

This paper proposes a new reconstruction based domain adaptation method for robust visual knowledge transfer. The method tends to reconstruct the target data with a few source data points by using a sparse coefficient matrix in some low-dimensional latent space. The method learns the sparse reconstruction coefficient matrix and the low-dimensional latent space projection simultaneously, such that an optimal subspace transfer solution can be obtained. Additionally, a kernel framework is generalized into this method, which aims at learning a nonlinear basis transformation and sparse reconstruction in the reproduced kernel Hilbert space induced by Mercer theorem, to deal with highly nonlinear domain shifts such as 3-D rotation of faces that cannot be tackled by linear techniques. Extensive experiments on synthetic data, two benchmark object datasets, Consumer & YouTube Videos datasets, CMU multi-PIE face dataset, and three handwritten digit datasets demonstrate the effectiveness of the proposed methods in different cross domain transfer tasks.

Appendix A
Optimization of (10)

With an auxiliary variable **L** and **U**, the problem (10) can be reformulated as

$$\begin{aligned}&\min_{\mathbf{Z},\mathbf{L}}\|\mathbf{Z}\|_1+\lambda_1\left\|\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}_T-\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\mathbf{L}\right\|_{\mathrm{F}}^2\\&\text{s.t.}\,\mathbf{L}=\mathbf{Z},\,\mathbf{1}_{N_S+N_T}^{\mathrm{T}}\mathbf{L}=\mathbf{1}_{N_T}^{\mathrm{T}},Z_{N_S+i,i}=U_i,U_i=0\end{aligned}\tag{15}$$

The augmented Lagrange function of (15) can be represented as

$$\begin{aligned}&J_3(\mathbf{Z},\mathbf{L},\mathbf{U})=\|\mathbf{Z}\|_1+\frac{\mu_1}{2}\left\|\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}_T-\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\mathbf{L}\right\|_{\mathrm{F}}^2+\langle\mathbf{Y}_A,\mathbf{L}-\mathbf{Z}\rangle\\&+\left\langle\mathbf{Y}_B,\mathbf{1}_{N_S+N_T}^{\mathrm{T}}\mathbf{L}-\mathbf{1}_{N_T}^{\mathrm{T}}\right\rangle+\sum_i Y_C^i\left(Z_{N_S+i,i}-U_i\right)+\langle\mathbf{Y}_D,\mathbf{U}\rangle\\&+\frac{\mu_2}{2}\left(\|\mathbf{L}-\mathbf{Z}\|_{\mathrm{F}}^2+\left\|\mathbf{1}_{N_S+N_T}^{\mathrm{T}}\mathbf{L}-\mathbf{1}_{N_T}^{\mathrm{T}}\right\|_2^2+\sum_i\left(Z_{N_S+i,i}-U_i\right)^2+\|\mathbf{U}\|_2^2\right)\end{aligned}\tag{16}$$

where $\mathbf{Y}_A$, $\mathbf{Y}_B$, $\mathbf{Y}_C$ and $\mathbf{Y}_D$ denote the Lag-multipliers, $\lambda_1=\mu_1/2$.

(1) Updating **L**: by fixing **Z** and **U**, one can set the partial derivative$\frac{\partial J_3(\mathbf{Z},\mathbf{L},\mathbf{U})}{\partial\mathbf{L}}=0$ of (16) as 0, and obtain **L** as

$$\begin{aligned}&\mathbf{L}=\left(\mu_1\mathbf{K}^{\mathrm{T}}\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}+\mu_2\mathbf{I}+\mu_2\mathbf{1}_{N_S+N_T}\mathbf{1}_{N_S+N_T}^{\mathrm{T}}\right)^{-1}\\&\left(\mu_1\mathbf{K}^{\mathrm{T}}\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}_T-\mathbf{Y}_A-\mathbf{1}_{N_S+N_T}\mathbf{Y}_B+\mu_2\mathbf{Z}+\mu_2\mathbf{1}_{N_S+N_T}\mathbf{1}_{N_T}^{\mathrm{T}}\right)\end{aligned}\tag{17}$$

(2) Updating **U**: by fixing **L** and **Z**, let$\frac{\partial J_3(\mathbf{Z},\mathbf{L},\mathbf{U})}{\partial U_i}=0$, we have

$$U_i=\frac{\mu_2 Z_{N_S+i,i}+Y_C^i-Y_D^i}{2\mu_2}\tag{18}$$

(3) Updating **Z**: by fixing **L** and **U**, **Z** can be solved as follows

i) for $Z_{N_S+i,i},\forall i=1,\dots,N_T$, we have

$$\begin{aligned}&\min\left|Z_{N_S+i,i}\right|+\frac{\mu_2}{2}\left(Z_{N_S+i,i}-L_{N_S+i,i}\right)^2-Y_A^{N_S+i,i}\cdot Z_{N_S+i,i}\\&+\frac{\mu_2}{2}\left(Z_{N_S+i,i}-U_i\right)^2+Y_C^i\cdot Z_{N_S+i,i}\\&\Leftrightarrow\min\left|Z_{N_S+i,i}\right|+\mu_2\left(Z_{N_S+i,i}-\left(\frac{L_{N_S+i,i}+U_i}{2}+\frac{Y_A^{N_S+i,i}+Y_C^i}{2\mu_2}\right)\right)\end{aligned}\tag{19}$$

ii) for **Z** other than $Z_{N_S+i,i},\forall i=1,\dots,N_T$, we have

$$\begin{aligned}&\min_{\mathbf{Z}}\|\mathbf{Z}\|_1+\frac{\mu_2}{2}\|\mathbf{L}-\mathbf{Z}\|_{\mathrm{F}}^2+\langle\mathbf{Y}_A,\mathbf{L}-\mathbf{Z}\rangle\\&\Leftrightarrow\min_{\mathbf{Z}}\|\mathbf{Z}\|_1+\frac{\mu_2}{2}\left\|\mathbf{Z}-\left(\mathbf{L}+\frac{\mathbf{Y}_A}{\mu_2}\right)\right\|_{\mathrm{F}}^2\end{aligned}\tag{20}$$

Appendix B
Proof of Proposition 2

The objective function of (11) can be expanded as follows

$$\begin{aligned}&\lambda_1\left\|\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\left(\mathbf{K}^{-1}\mathbf{K}_T-\mathbf{Z}\right)\right\|_{\mathrm{F}}^2+\lambda_2\left\|\mathbf{X}\left(\mathbf{I}-\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\right)\right\|_{\mathrm{F}}^2\\&=Tr\left(\lambda_1\left(\mathbf{K}^{-1}\mathbf{K}_T-\mathbf{Z}\right)\left(\mathbf{K}^{-1}\mathbf{K}_T-\mathbf{Z}\right)^{\mathrm{T}}\mathbf{K}^{\mathrm{T}}\mathbf{Q}^{\mathrm{T}}\mathbf{K}\right)\\&\qquad+Tr\left(\lambda_2\left(\mathbf{K}-2\mathbf{K}^{\mathrm{T}}\mathbf{Q}^{\mathrm{T}}\mathbf{K}+\mathbf{K}^{\mathrm{T}}\mathbf{Q}^{\mathrm{T}}\mathbf{K}\mathbf{Q}\mathbf{K}\right)\right)\end{aligned}\tag{21}$$

where $\mathbf{Q}=\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}\in\Re^{N\times N}$.

By following (21), with $\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\mathbf{\Phi}=\mathbf{I}$, there is $\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}\mathbf{K}\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}=\mathbf{Q}\mathbf{K}\mathbf{Q}^{\mathbf{T}}=\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}}=\mathbf{Q}$, then the objective (21) can be simplified as

$$Tr\left(\left(\lambda_1\left(\mathbf{K}^{-1}\mathbf{K}_T-\mathbf{Z}\right)\left(\mathbf{K}^{-1}\mathbf{K}_T-\mathbf{Z}\right)^{\mathrm{T}}-\lambda_2\mathbf{I}\right)\mathbf{K}^{\mathrm{T}}\mathbf{Q}^{\mathrm{T}}\mathbf{K}\right) \tag{22}$$

According to the Eigenvalue decomposition of $\mathbf{K}=\mathbf{VSV}^{\mathrm{T}}$ [5], we obtain $\mathbf{K}^{\mathrm{T}}\mathbf{Q}^{\mathrm{T}}\mathbf{K}=\mathbf{VS}^{\frac{1}{2}}\mathbf{\Omega\Omega}^{\mathrm{T}}\mathbf{S}^{\frac{1}{2}}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{\Omega}=\mathbf{S}^{\frac{1}{2}}\mathbf{V}^{\mathrm{T}}\mathbf{\Phi}$.

Then the objective function (22) can be rewritten as

$$Tr\left(\mathbf{\Omega}^{\mathrm{T}}\mathbf{S}^{\frac{1}{2}}\mathbf{V}^{\mathrm{T}}\left(\lambda_1\left(\mathbf{K}^{-1}\mathbf{K}_T-\mathbf{Z}\right)\left(\mathbf{K}^{-1}\mathbf{K}_T-\mathbf{Z}\right)^{\mathrm{T}}-\lambda_2\mathbf{I}\right)\mathbf{VS}^{\frac{1}{2}}\mathbf{\Omega}\right)=Tr\left(\mathbf{\Omega}^{\mathrm{T}}\mathbf{\Theta\Omega}\right) \tag{23}$$

where $\mathbf{\Theta}=\mathbf{S}^{\frac{1}{2}}\mathbf{V}^{\mathrm{T}}\left(\lambda_1\left(\mathbf{K}^{-1}\mathbf{K}_T-\mathbf{Z}\right)\left(\mathbf{K}^{-1}\mathbf{K}_T-\mathbf{Z}\right)^{\mathrm{T}}-\lambda_2\mathbf{I}\right)\mathbf{VS}^{\frac{1}{2}}$ and $\mathbf{\Omega}^{\mathrm{T}}\mathbf{\Omega}=\mathbf{\Phi}^{\mathrm{T}}\mathbf{VSV}^{\mathrm{T}}\mathbf{\Phi}=\mathbf{\Phi}^{\mathrm{T}}\mathbf{K\Phi}=\mathbf{I}$.

Finally, the original optimization problem (11) becomes

$$\mathbf{\Omega}^{*}=\arg\min_{\mathbf{\Omega}} Tr\left(\mathbf{\Omega}^{\mathrm{T}}\mathbf{\Theta\Omega}\right), \text{s.t.}\ \mathbf{\Omega}^{\mathrm{T}}\mathbf{\Omega}=\mathbf{I} \tag{24}$$

The optimal $\mathbf{\Omega}^*$ is obtained by l eigenvectors with respect to the first l smallest Eigenvalues of $\mathbf{\Theta}$. Once $\mathbf{\Omega}^*$ is solved by $\mathbf{\Omega}=\mathbf{S}^{\frac{1}{2}}\mathbf{V}^{\mathrm{T}}\mathbf{\Phi}$ and $\mathbf{VV}^{\mathrm{T}}=\mathbf{I}$, the optimal $\mathbf{\Phi}^*$ can be solved as

$$\mathbf{\Phi}^{*}=\mathbf{VS}^{-\frac{1}{2}}\mathbf{\Omega}^{*} \tag{25}$$

ACKNOWLEDGMENT

We are grateful to the AE and anonymous reviewers for their valuable comments on our work.

REFERENCES

[1] M. Shao, D. Kit, and Y. Fu, "Generalized Transfer Subspace Learning Through Low-Rank Constraint," Int. J. Comput. Vis., vol. 109, pp. 74-93, 2014.

[2] I.H. Jhuo, D. Liu, D. Lee, and S.F. Chang, "Robust visual domain adaptation with low-rank reconstruction," CVPR, pp. 2168-2175, 2012.

[3] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 11, pp. 2675-2781, 2013.

[4] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," ICML, pp. 663-670, 2010.

[5] A.H. Sameh and J.A. Wisniewski, "A trace minimization algorithm for the generalized eigenvalue problem," SIAM J. Numer. Anal., vol. 19, no. 6, pp. 1243-1259, 1982.

[6] S. Shekhar, V.M. Patel, H.V. Nguyen, R. Chellappa, "Generalized Domain-Adaptive Dictionaries," CVPR, pp. 361-368, 2013.

[7] H. Daumé, "Frustratingly easy domain adaptation," ACL, vol. 45, pp. 256-263, 2007.

[8] R. Gopalan, R. Li, and R. Chellappa, "Domain adaptation for object recognition: An unsupervised approach," ICCV, 2011.

[9] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new Domains," ECCV, 2010.

[10] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," CVPR, pp. 2066-2073, 2012.

[11] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Institute of Technology: Tech.rep, 2007.

[12] J. Yang, R. Yan, and A. Hauptmann, "Cross-domain video concept detection using adaptive SVMs," ACM MM. 2007.

[13] V.M. Patel, H.V. Nguyen, and R. Vidal, "Latent Space Sparse Subspace Clustering," ICCV, 2013.

[14] J. Hoffman, E. Rodner, J. Donahue, K. Saenko, and T. Darrell, "Efficient Learning of Domain Invariant Image Representations," ICLR, 2013.

[15] B. Kulis, K. Saenko, and T. Darrell, "What you saw is not what you get: Domain adaptation using asymmetric kernel transforms," In Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition, pp. 20-25, June 2011.

[16] L. Duan, D. Xu, W. Tsang, and J. Luo, "Visual Event Recognition in Videos by Learning from Web Data," IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 9, pp. 1667-1680, Sep. 2012.

[17] L. Duan, W. Tsang, and D. Xu, "Domain Transfer Multiple Kernel Learning," IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 3, pp. 465-479, Mar. 2012.

[18] L. Duan, D. Xu, and I. Tsang, "Domain adaptation from multiple sources: A domain-dependent regularization approach," IEEE Trans. Neural Networks and Learning Systems, vol. 23, no. 3, pp. 504-518, 2012.

[19] J. Hoffman, E. Rodner, J. Donahue, B. Kulis, K. Saenko, "Asymmetric and Category Invariant Feature Transformations for Domain Adaptation," Int. J. Comput. Vis., vol. 109, pp. 28-41, 2014.

[20] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 1, pp. 171-184, Jan. 2013.

[21] E. Elhamifar and R. Vidal, "Sparse subspace clustering," CVPR, pp. 2790-2797, 2009.

[22] Z. Lin, M. Chen, L. Wu, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," Technical Report, UILU-ENG-09-2215, 2009.

[23] R. Gross, I. Matthews, J.F. Cohn, T. Kanade, and S. Baker, "Multi-pie," Image Vision Computing, vol. 28, no. 5, pp. 807-813, 2010.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.

[25] A. Frank and A. Asuncion, (2010) UCI machine learning repository [Online]. Available: http://archive.ics.uci.edu/ml.

[26] S. Boyd, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," Foundations and Trends in Machine Learning, 2010.

[27] A.C. Loui, J. Luo, S.F. Chang, D. Ellis, W. Jiang, L. Kennedy, K. Lee, and A. Yanagawa, "Kodak's Consumer Video Benchmark Data Set: Concept Definition and Annotation," *Proc. Int'l Workshop Multimedia Information Retrieval*, pp. 245-254, 2007.

[28] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning Realistic Human Actions from Movies," *CVPR*, pp. 1-8, 2008.

[29] D.G. Lowe, "Object recognition from local scale-invariant features," *ICCV*, pp. 1150-1157, 1999.

[30] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," Int'l J. Computer Vision, vol. 60, no. 2, pp. 91-110, 2004.

[31] M. Belkin and P. Niyogi, "Semi-supervised learning on manifolds," NIPS, 2002.

[32] A. Blum and S. Chawla, "Learning from Labeled and Unlabeled Data using Graph Mincuts," ICML, 2001.

[33] S.J. Pan, I.W. Tsang, J.T. Kwok, and Q. Yang, "Domain Adaptation via Transfer Component Analysis," IEEE Trans. Neural Networks, vol. 22, no. 2, pp. 199-210, Feb 2011.

[34] S.J. Pan and Q. Yang, "A survey on transfer learning," IEEE Trans. Knowl. Data Eng., vol. 22, no. 10, pp. 1345-1359, Oct. 2010.

[35] J. Ghosn and Y. Bengio, "Bias learning, knowledge sharing," IEEE Trans. Neural Networks, vol. 14, no. 4, pp. 748-765, Jul. 2003.

[36] L. Zhang and D. Zhang, "Domain Adaptation Extreme Learning Machines for Drift Compensation in E-Nose Systems," IEEE Trans. Instrumentation & Measurement, vol. 64, no. 7, pp. 1790-1801, 2015.

[37] M. Soltanolkotabi, E. Elhamifar, and E.J. Candès, "Robust Subspace Clustering," Ann. Statist., vol. 42, no. 2, pp. 669-699, 2014.

[38] A. Krizhevsky, I. Sutskever, G.E. Hinton, "ImageNet classification with deep convolutional neural networks," NIPS, 2012.

[39] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition," ICML, 2014.

[40] L. Zhang and D. Zhang, "Robust Visual Knowledge Transfer via EDA," arXiv: 1505.04382, 2015.

[41] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised Visual Domain Adaptation Using Subspace Alignment," ICCV, pp. 2960-2967, 2013.

[42] M. Baktashmotlagh, M.T. Harandi, B.C. Lovell, and M. Salzmann, "Unsupervised Domain Adaptation by Domain Invariant Projection," ICCV, pp. 769-776, 2013.

[43] X. Peng, L. Zhang, and Y. Zhang, "Scalable Sparse Subspace Clustering," CVPR, pp. 430-437, 2013.

[44] L. Lin, Y. Xu, X. Liang, and J. Lai, "Complex Background Subtraction by Pursuing Dynamic Spatio-Temporal Models," IEEE Trans. Image Processing, vol. 23, no. 7, pp. 3191-3202, Jul 2014.